ON-LINE, REAL-TIME, MULTIPLE SPEECH OUTPUT SYSTEM AND ITS SYSTEM EVALUATION

by

Shinji TOMITA

Department of Information Science

Kyoto University

November, 1973

ON-LINE, REAL-TIME, MULTIPLE SPEECH OUTPUT SYSTEM AND ITS SYSTEM EVALUATION

SHINJI TOMITA

Department of

Information Science

Kyoto University

November, 1973

ACKNOWLEDGEMENTS

The author wishes to express his sincere thanks to Professor Toshiyuki Sakai for his encouragement and valuable suggestions during the course of this study. He also wishes to thank Research Associate Kenji Ohtani for his invaluable support in the experiments. Many other people assisted in this study. The author is grateful to Professor Makoto Nagao, Associate Professors Shigeharu Sugita and Yasuhisa Niimi, Research Associates Koichi Tabata, Takeo Kanade and Susumu Yoshida, Graduate Course Students Susumu Yamazaki, Shigeyoshi Kitazawa, Koichi Ohnishi, Hitoshi Aoki and members of Professor Sakai's research group for helpful discussions and co-operation. The author is also indebted to Professor Hiroshi Hagiwara for his helpful support and accommodation. The author offers his most heartfelt appreciation to his wife Kazuko who always cheers him up.

CONTENTS

I Introduction

This thesis is concerned with an automatic speech output system developed by the author during the doctor course at Kyoto Univ..

Chapter I introduces a historical review on automatic speech production and queueing theory. This may be helpful for comprehension of this thesis.

Chapter II describes the general configuration and some behaviors of the proposed system. It shows how to produce multiple outputs of arbitrary speech sounds automatically by a computer.

Key concepts for the automatic speech production are; (1) a computer program compiles required speech elements successively, which are pre-edited and stored on a magnetic drum in a numerical form of zero-crossing waves. (2) each speech synthesizer converts digital data of a speech element sent from the computer into an analog form and produces an independent output of speech sounds.

Chapter III presents several queueing models of the system and the related problems of the system evaluation. It discusses how many independent speech outputs are available. Various types of the processing schemes are presented and the results obtained by simulation of them are compared for the purpose of improving system performance.

Chapter IV discusses special problems of queueing control on the speech output system. A proposed scheme

controls requests before their arrivals to the system and succeeds in keeping the system from occasional congestion. Results of simulation show that the scheme improves the system performance to some extent.

Chapter V sums up the results of this study and shows some unsolved problems for further research.

Symbols and notations used are listed at the end of the pages for convenient reference.

1.1 Various Types of Automatic Speech Output System

In the rapid progress of information science and technology, conversational operation between man and machine has become essential and powerful in many fields of computer applications.

For interactive communications with the computer, a wide variety of peripheral units have been developed. These are classified into two groups. One is through visual images which includes graphic displays, light pens, etc. The other is through auditory information which includes speech input and output devices.
Each device has its own features and has been in practical use effectively for some particular applications.

Graphic displays and light pens are now widely used in such an increasing application as CAD (computer aided design).

Speech input to the computer is related to one of the special fields of pattern recognition which is still

in general far from being successful. The difficulty in speech recognition lies mainly in the fact that natural speech has too wide a range of possible patterns and inherent indeterminacy to be identified by the computer.

On the other hand, speech output is not so difficult as speech input. Various types of speech output systems have been developed and small versions of them such as IBM 7772 audio response unit[1] have become popular in commercial applications. They are now also planned to be used as audio response units even in seat reservation systems through push phones in the railway systems of J.N.R.(Japan National Railway)[2].

Speech is one of the most powerful means for communication. It is not only an economical transport of much information, but also has redundant information so much as to be audible even if unexpected noise is superimposed on it. These intrinsic characteristics are very important aspects in effective man-machine communication. Suppose that an operator communicates with a computer only through visual display units. It seems awfully tiresome and time-consuming. It is no exaggeration to say that a speech output system, even in a small version, will be indispensable for the convenient use of computers.

Main attributes of speech output systems required for popular use[3,4] are; (1) output of speech sounds of high quality, (2) output of arbitrary messages (i.e.,infinite vocabulary), (3) simultaneous outputs to many users, and (4) economical implementation and easy connection to computers.

Until now, many investigators have tried to implement systems which may satisfy all of the requirements mentioned above, but they achieved only partial

success because the systems have conflicting factors still unsolved in quality of speech sounds and/or in the problem of system designing.

Among various kinds of speech output systems, voice-recording-reproducing type was first reported in 1963 by Lee et al.[5]. This is the simplest because it only reads out pre-recorded voice of each word successively and then connects them into continuous speech sounds. Each word is collected from natural speech and stored on a secondary memory in advance either in an analog or a digital form. Resulting speech sounds are necessarily very high in quality, because of direct output of natural speech. Simultaneous outputs to many users are easily realized by using either 2-speed buffers or multiplexor channels. On the other hand, output vocabulary is severely limited to 128-256 words because of the enormous memory size required for storing data of the words. This type of system is simply implemented and then often used in special applications where a small vocabulary of words is sufficient. Modified speech output systems of this type have been reported, one of which is used in DIALS (Dendenkosha Immediate Arithmetic & Library System)[6].

Other methods of automatic speech output have been also developed to remove severe restriction of the small output vocabulary. They are practically based either on synthesizing speech sounds from sequence of some intrinsic parameters or on connecting smaller speech elements successively into a continuous form. This speech synthesis method reduces the amount of stored information necessary for producing speech sounds. The following are some typical examples of this type:

(1) Vocoder type speech synthesis method

This was made by using some technical applications of Vocoder, and now on sale as IBM 7772 audio response unit.[1] This system can produce speech sounds with a limited vocabulary of about 1000 words. The necessary amount of information for controlling the speech synthesizer is 2.4 kbits/sec. The memory size required for storing data is reduced by 1/20 as compared with that of voice-recording-reproducing type.

Recently, Itakura et al.[7] proposed PARCOR speech synthesis method which was analogous to the Vocoder type. It produces multiple speech outputs of about 10 channels from particular parameter string called PARCOR(Partial Auto-Correlation Coefficient).

(2) Terminal analog speech synthesis method[8,9,10]

This method simulates the overall transmission characteristics of the vocal tract. The transfer function of the vocal tract has large number of poles and zeros, among which the lower two or three play an essential role for speech production. The speech synthesizer continuously controls wave intensity, pitch frequency and parameter values of formants (poles) by means of resonance circuits and gets the resulting speech sounds. Quality of the output speech sounds is rather high when the set of parameter of good quality is selected context-wise by the cut-and-try method. But, if the parameter is determined automatically only by rules, output speech is of less quality, while the system can produce arbitrary messages. The necessary amount of controlling information is about 2.4 kbits/sec.

(3) Vocal tract analog speech synthesis method

The serially concatenated tubes are approximate models for structure of the vocal tract. In order to simulate this multi-tube system, digital and analog computer simulations have been conducted. Recent research has been centered on the latter, which enables continuous control on the equivalent electrical lump circuits for the multi-tubes. One of this type is DAVO[11] made recently by the research group in ETL (Electrotechnical Laboratory of Japan). But, output speech is not so good in quality even if the precise controlling parameters are given.

In all of these systems on speech synthesis method, multiple speech output requires many independent speech synthesizers controlled by a device attached to a computer. The construction of many synthesizers is a great expense.

(4) Speech synthesis method by the compilation of acoustical elements of speech

Recently, Nakata et al.[12] reported speech output system based on the mixture of both voice-recording-reproducing method and the speech synthesis method. Speech elements consist of many kinds of damped sinusoids within one pitch period and band pass noises. They are stored on a magnetic drum. If a controlling parameter sequence is given, some of the speech elements are read out syncronously from the magnetic drum and merged together into resulting speech sounds. They said that the system could produce speech sounds of unlimited vocabulary of words and serve more than 100 users simultaneously. But, the output speech is not so complete for commercial applications. Matsui et al.[13] also presented a similar speech

synthesis method, but have not yet constructed it.

A new speech output system discussed in Chapter II belongs to the speech synthesis method. The speech elements consist of pulse sequence of zero-crossing intervals of waves within one cycle at larynx frequency for voiced sounds. Sakai and Ohtani previously reported that the speech output system of this type could produce any kind of speech sounds with high intelligibility and in real-time mode.[14,15,16] The proposed system is an extended version of the system by Sakai and Ohtani, being able to produce multiple outputs of speech. Of course, it requires the same number of synthesizers as the users to be served. However, since its organization is quite simple, the system's synthesizers are not so costly to construct.

1.2 Queueing Models and System Evaluation

With miraculous development of larger scale computers, system evaluation on them has played an important role in systems designing. In order to utilize the resources efficiently, they must not only be analyzed quantitatively at the designing stage, but also be verified at the operating stage that they satisfy all of the specifications, running without any unexpected bottlenecks.

Until now, various kinds of technical methods for evaluating system performance have been developed. These include; (1) mathematical analysis, (2) Monte Carlo

method, and (3) hardware and software monitoring.

To construct a mathematical model, some important parameters must be abstracted from the system to be analyzed. One of the powerful analytical tools is the queueing theory to describe dynamic behaviors of the system.

A queueing operation is divided into four parts; the input, the waiting line, the service facility and the output. With each of these is associated a set of alternative assumptions concerning the queueing process, some of which have been the object of research in the field.

Saaty showed these variations listed in Table 1-1 and collected about a thousand of papers up to 1961 on the queueing theory and its applications.[17]

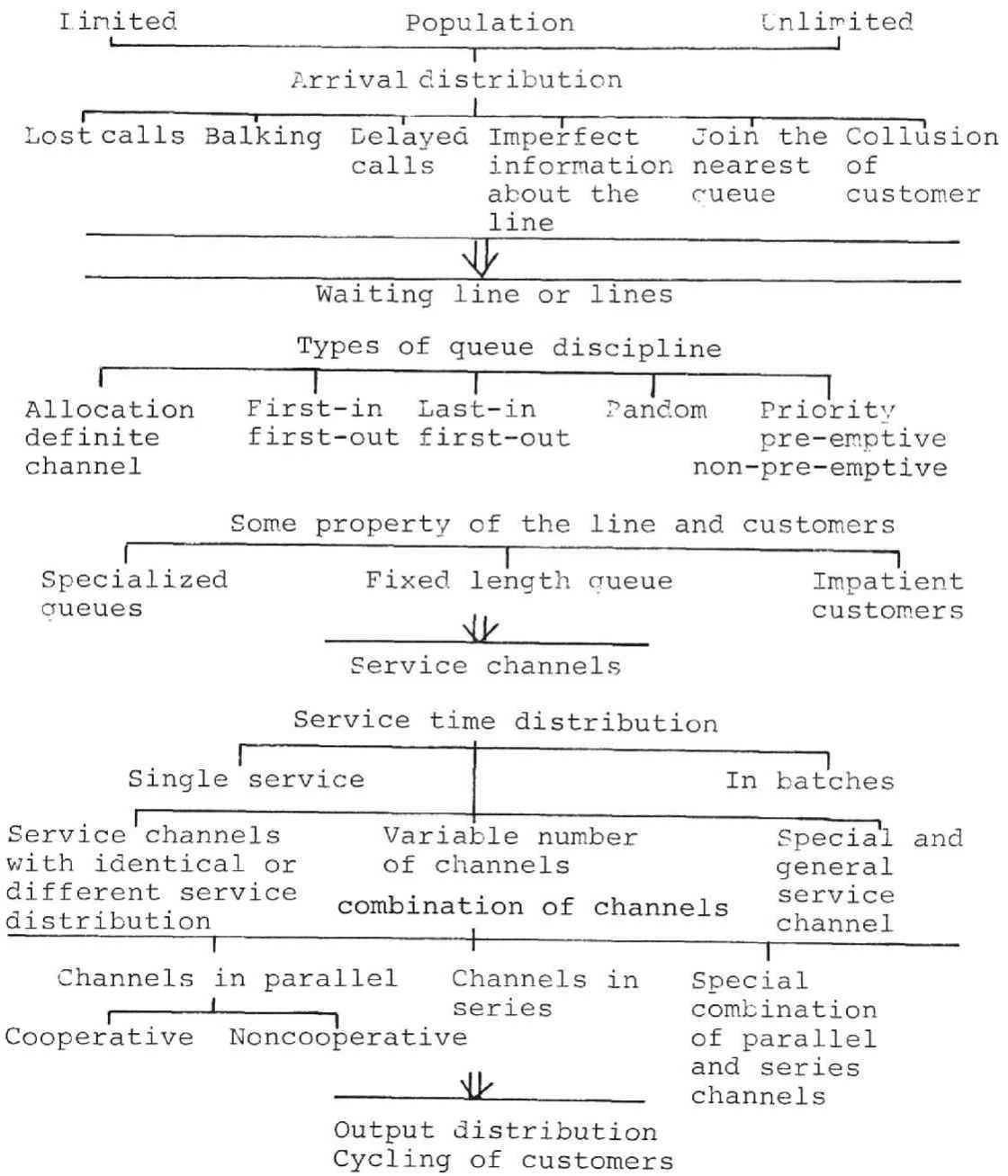
But, in the case of a complicated model, it is almost impossible to solve analytically, even if the model is approximately simplified, because of the formidable states needed to describe the model in a precise manner. Mathematical analysis on the approximate model is,however, often available to show the qualitative insights into the systems for designers.

The Monte Carlo method is one of the most powerful tools for the analysis of stochastic processes that can hardly be solvable in a mathematical way. But, it is a time-consuming way to get meaningful results in the equilibrium state.

It is very important for further considerations to collect information on dynamic behaviors in complicated computer systems. For this purpose, monitoring techniques[19,20] are very useful. Special devices have been constructed in hardware and computer complex in order to collect data directly from CPU and/or I/O devices with as little

Table 1-1

Variation of the queueing process



disturbance as possible. These data would be presented and analyzed in an empirical way by system designers. At the present technical level, system evaluation on highly complicated systems still remains to be heuristic.

Chapter III not only discusses system evaluation by monitoring method on the multiple speech output system, but also presents various techniques for improving performance of the system.

In the system, various kinds of queues are formed when many synthesizers are supposed to be connected to the computer. While, only one CPU installed in the computer executes many kinds of processings in an order of pre-assigned priority. A precise queueing model of the system is classified as a moving server queueing model that is too complicated to be solvable. Therefore, it is most realistic to deal with the simplest model containing the system's bottleneck points and to show the way to break them for the improvement of system performance. Results of simulation in real-time mode are considered in those aspects such as the load effect of the computer, maximum number of multiplicity and the required size of memory.

One of the interesting problems for improvement of system performance is closely related to controlling the queueing process. In on-line, real-time systems many kinds of queues are formed. Requests occur at random and contend with each other to occupy the processing units. But, in any cases, processing units are limited in their capability. Therefore, it is essential to supervise those queues and control them in a suitable manner, in order to utilize efficiently all of the resources within the system

and/or increase the speed of real-time processings.

One of the typical examples is a model of a taxi terminal.[21,22,23] In this model, a passenger arrives at random at the terminal, and would wait in the queue if he expected from the present queue length that he could take a taxi in a few minutes. But, if the queue length is too long, he would not enter the queue. Queueing control may be done to improve profits of the taxi company in such a manner as to suppress a number of balking (not entering) passengers by increasing the service rate. Here, the service rate corresponds to the arrival rate of a taxi to the terminal.

The same controlling method would be adopted in large scale computer systems which operate on both the real-time and batch processings. If necessary, as in the case of the task scheduling and dispatching, a monitor of a computer system allocates much more CPU time to the particular processings.[24]

But, in the two examples mentioned above, the input process still remains uncontrolled. In general, however, it is very difficult to regulate incoming requests before arrivals in some order, because of the random arrivals from the infinite population. But, in special cases, controlling the input process would be possible. One of the typical examples is as follows. Suppose the case of traffic control on a network of roads. If many cars parking at the parking area move and enter the network of the roads at one time, traffic in the network will become jammed and get out of hand. But, if they are regulated to go out at the exit of the parking area, then the traffic entering the queueing network (each queue is formed at the signal station) is controlled to suppress excessive load on the network.

The same concept is introduced in the computer network which remarkable notice has been paid to recently.[25,26] The control is often referred to as a flow control. The adaptive routing algorithm and communication procedures (e.g.,communication on a logical link) in the ARPA computer network[45,46] enable the whole system to be operated effectively. In order to prevent congestion in the network, the network accepts only one message at a time on a given logical link. Ensuring messages on that link will be blocked from entering the network until the previous message has arrived at the destination.

A familiar example of this controlling input process is reservation of diagnostic examination time in a hospital. Input process is scheduled so that patients usually need not wait much longer than before, as long as an emergency case does not occur.

For effective operations of the system, scheduled arrivals will have to be introduced to some extent. This means that it is fundamentally important to prevent the overload and underload conditions, by removing the random events as much as possible. But, this sort of problem remains still almost unsolvable because of a too complicated model for mathematical analysis.

Chapter IV is devoted to the description of particular problems of the queueing control on the speech output system. A structure of the system implemented to realize control scheme is analogous to the illustrative model as shown in Fig. 1-1. A computer supplies a buffer of a device with some amount of data. The device consumes these data at a certain rate. At the time when the buffer becomes empty, the device requests the computer to transmit the next data. If other devices' requests are congested in the system, it may happen that the buffer

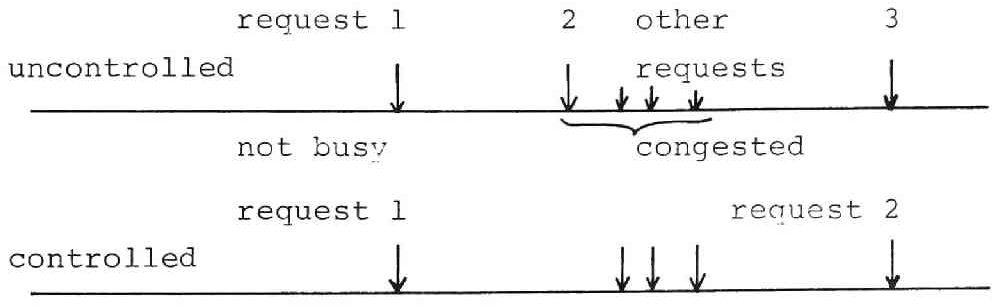


Fig. 1-1 Illustrative model under queueing control

remains empty for a long time. In order to prevent this situation, the computer forecasts a coming congestion when it is not busy and supplies the buffer with much longer data than usual. This control scheme prevents a request from occurring at the time when it would occur. The total capacity to store data would be controlled to be as small as possible.

The proposed scheme controls requests before their arrivals to the system and succeeds in keeping the system from occasional congestion. Results of simulation show that queueing control practically improves the system performance to some extent, particularly on the maximum number of simultaneous outputs of speech sounds.

II Proto-type of On-line,Real-time,Multiple Speech Output System[27.28.29.30]

2.1 Introduction

This chapter describes the general configuration of the system and some evaluation upon the system performance from the statistical point of view.

Key concepts and characteristics of the system are;
(1) A computer program compiles required speech elements successively, which are pre-edited and stored on a magnetic drum in a numerical form of zero-crossing waves.
(2) Each speech synthesizer converts digital data of a speech element sent from the computer into an analog form and produces a corresponding output of speech sounds.
(3) Simultaneous outputs of arbitrary messages can be produced in on-line,real-time mode.
(4) The synthesized speech sounds are not so natural as that of voice-recording-reproducing type. However, they have enough intelligibility for human listening.
(5) In comparison with other systems, notable benefits of the system are economical implementation and a smaller memory space for storing all of the speech elements.

In the on-line, real-time system, many processing requests for a central processor (CPU) and/or I/O devices, may occur at random. They are set in queues to wait for processing until these devices are available. If so long

a queue was made in line that a waiting time was longer than permitted for a request, the system could not process in real-time mode. Therefore, it is very important to know the various statistical data in the multiple speech output system on which the load effect of the computer and the maximum number of simultaneous outputs are strongly dependent. Another important aspect of statistical analysis is to pick up systems' bottleneck points and to show the way to break them for system improvement.

2.2 Speech Synthesis by Using Zero-crossing Wave

Zero-crossing wave is a pulse sequence with time variant intervals. It can be produced by infinitely amplifying the original speech sound and then clipping the amplitude at a certain level (infinite peak clipping). Fig.2-1 shows an example of zero-crossing wave. It was Licklider[31,32,33] that made first systematic experiments to get the intrinsic characteristics of zero-crossing wave. The wave produced from a natural wave through cascaded operations of differentiation, infinite peak clipping and integration still contained some important information through which the linguistic contents of the wave may be similarly identified for human listening. The experiments showed that the first and the second formant loci in zero-crossing wave were quite similar to those of natural wave. The score of word-articulation test on zero-crossing wave was up to about 97%.

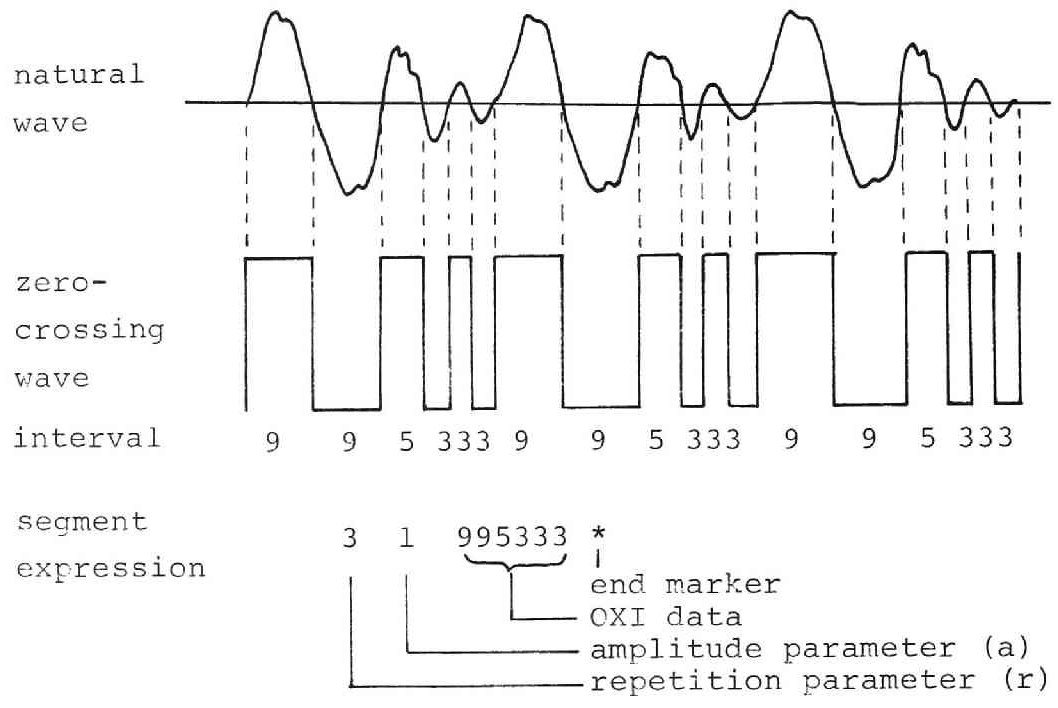


Fig. 2-1 Zero-crossing wave

These surprising facts formed the experimental basis on which zero-crossing wave was found out to be available for speech synthesis. Furthermore, it was expected that a speech output system could be made economically because controlled data were in a digital form for which computer and digital techniques could be easily applied.

Primary elements adopted for compilation in the system are called wave elements (abbreviated as "P"). They consist of pulse sequence of zero-crossing intervals which correspond to waves within one cycle at larynx frequency for voiced sounds.

It is well known that the voiced sounds like vowels, nasals, and voiced plosives have in their waveforms a sequence of quasi-periodic patterns repeated at the frequency of vocal cord vibration. This means that those

voiced sounds can be expressed simply with a typical sequence of zero-crossing intervals (henceafter abbreviated as OXI) and two associated parameters; the number (r) of repetitions necessary to reproduce the original length of the voiced sounds and the wave amplitude (a). The wave element associated with two parameters is called "segment" which is denoted as " aP^{r} ". For example, Fig.2-1 shows zero-crossing wave and its numerical expression by a segment. By this principle, each voiced sound is expressed in a simple form, particularly suitable for storing data and processing by digital equipments.

(1) Segment expression of vowel and consonant parts:

Several phonemes expressed by segment sequence are listed in Table 2-1.

Table 2-1

Expression of phonemes with a segment sequence

phoneme	expression	phoneme	expression
/a/	$17P_a^8$	/d/	$3P_d^5 \cdot 3P_t^1$
/e/	$13P_e^8$	/g/	$3P_g^5 \cdot 3P_k^1$
/m/	$3P_m^7$	/s/	$1P_s^{10}$ *
/n/	$3P_n^7$	/z/	$2P_{z1}^4 \cdot 2P_{z2}^2 \cdot 2P_s^4$ *
/p/	$0P_{sp}^9 \cdot 2P_p^1$	/ch/	$0P_{sp}^7 \cdot 2P_{ch}^6$ *
/t/	$0P_{sp}^9 \cdot 2P_t^1$	/ts/	$0P_{sp}^7 \cdot 2P_{ts}^6$ *
/k/	$0P_{sp}^9 \cdot 2P_k^1$	/h/	$1P_h^2$ *
/b/	$3P_b^5 \cdot 3P_p^1$	/r/	$2P_r^5$

The duration of the retention of a vowel, which is about 70 msec in the normal context, can be changed by controlling the value of the repetition parameter (r)

of the segment.

Nasal sounds /m/ and /n/ are also produced by the repetitions of wave elements P_m and P_n respectively. The unvoiced stop consonants /p/, /t/, and /k/ consist of the wave elements P_p , P_t , and P_k preceded by a silent interval P_{sp} respectively. Voiced stops /b/, /d/, and /g/ are constructed with five repetitions of buzz parts P_b , P_d and P_g , followed by burst parts P_p , P_t , and P_k , respectively, which are identical with the wave elements used in the unvoiced stops /p/, /t/ and /k/.

In the expressions of /s/, /z/ and so on, found in Table 2-1, which have relatively long-continued noisy periods, the number of repetitions is accompanied by an asterisk (*) as is shown in the case of /s/.
The expression 10* of $1P_s^{10*}$ does not mean that the wave element P_s is simply repeated 10 times, but, that the sequence of OXI of P_s is connected 10 times, being re-arranged at random at every connection. Wave elements P_{ch} and P_{ts} in affricative sounds /ch/ and /ts/ are processed in the same manner. The reason for this is that those sounds have no periodic patterns.

All of the wave elements listed in Table 2-1 are extracted from Japanese monosyllabic sounds uttered by a male. A speech sound wave is pre-emphasized with an RC-high pass filter (6 db/octave) whose cut-off frequency is 1.6 kHz, and is sampled at 20 kHz rate. The amplitude is quantized into 10 bits plus a sign bit with an A/D converter connected to the computer and is subjected to the infinite peak clipping by a computer program. The OXI patterns are selected by hand from the retentive parts. Only 32 of 550 wave elements are extracted in this way. Others are produced by a computer program as follows:

(2) Segment interpolation in the glide parts:

When two different phonemes are uttered continuously, articulatory organs move necessarily under some physical constraints. In the glide part corresponding to this transitional movement, the appropriate segment sequence must be interpolated.

In a V-C-V (V:vowel,C:consonant) context, the second formant frequency disappears at the offset point in the transition part from a vowel to a consonant, and again clearly appears at the onset point from a consonant to a vowel. Formant frequencies at these points are closely related to each other, but depend only on a particular V-C-V context.[34] The offset and onset formant frequencies can be determined empirically by the inspection of the patterns of sonagrams. Such a sequence of the segments is interpolated in a glide part that formant loci from a vowel part to an offset point, or from an onset point to a vowel part are linearly moved. And also, amplitude parameter of each interpolated segment is modified to give continuous envelope of synthesized waves.

Since it is difficult to select the wave elements of the glide part from natural speech for all of the contexts, these wave elements are produced by a computer simulation of the terminal analog speech synthesis model, which consists of the series concatenation of 4 single tuned circuits (S.T.C). The center frequencies f_1 and f_2 of the lower two S.T.C. correspond to the frequencies of the first and the second formants.Here,the center frequency f_1 is varied by 50 Hz from 300 Hz to 950 Hz and f_2 from 700 Hz to 2500 Hz. The center frequencies of the third and fourth S.T.C. are fixed at constant values.
For each pair of f_1 and f_2 parameters, the simulation program is executed. The resulting sound wave produced for

one steady larynx cycle is transformed into a zero-crossing wave, and is registered as the wave element data.

Some of these wave elements are selected to interpolate the glide part of particular contexts. For example, V-C-V context of "oda" is transformed into the segment sequence as follows.

$$10P_{o}^{8} \cdot 9P_{1(550,1100)}^{1} 8P_{2(500,1200)}^{1} 6P_{3(450,1300)}^{1} 3P_{d}^{5} \cdot 3P_{t}^{1}$$
$$9P_{1(400,1450)}^{1} 12P_{2(600,1350)}^{1} 15P_{3(750,1250)}^{1} 17P_{a}^{8}$$

Here the notation $P_{1(400,1450)}^{1}$,etc. means a computed wave element of one-pitch period in the glide part whose first and second formants are 400 Hz and 1450 Hz respectively. Fig.2-2 illustrates a part of the synthesized speech obtained from the above segment sequence schematically.

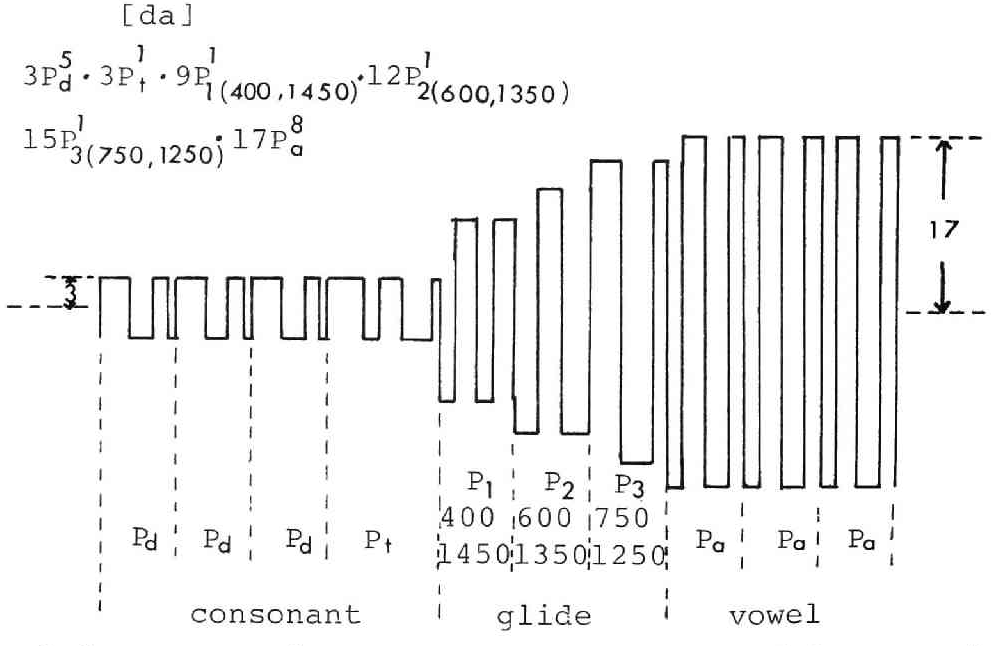


Fig. 2-2 Expression of the syllable [da] with segment sequence and its synthesized wave form

2.3 General Description of the System

2.3.1 Actual Compilation Element

In the previous section, definitions of the wave element and the segment for speech synthesis were given. Although it requires a smaller memory space of about 70 kbits to store all of 550 wave elements on the magnetic drum, the operation for reading out and connecting the stored wave elements according to an editing rule by a computer program will be too time-consuming for the multiple simultaneous speech outputs.

The actually adopted set of compilation elements consists of segment sequence corresponding to about 180 Japanese consecutive phonemes such as vowel-consonant (V-C), consonant-vowel (C-V) and vowel-vowel (V-V) contexts. Henceafter, these segment sequences are referred to as " dyad segment sequences ". These are pre-edited and stored on the magnetic drum in a numerical form of zero-crossing wave. The glide parts of dyad segment sequences of (C-V),(V-C) and (V-V) contexts, are interpolated by the method mentioned before (see," Segment interpolation in the glide parts ", in Section 2.2). Here, either vowel preceded or followed by a consonant in V-C-V context, is fixed to a neutral vowel whose first and second formant frequencies are 500 Hz and 1500 Hz respectively.

A dyad segment sequence for (C-V) stored on the magnetic drum, consists of segment sequences to both consonant and glide parts. While, from a dyad segment sequence for (V-C) or (V-V) a vowel part is excluded. It is stored residently in the core memory of the computer.

The reasons for this are; (1) to reduce the necessary memory space on the magnetic drum; (2) to permit a vowel part to be modified there in the case when that vowel is accented. In the monosyllabic sound [da] shown in Fig.2-2, the segment sequence $3P_d^5 \cdot 3P_t^1 \cdot 9P_{1(400,1450)}^1 \cdot 12P_{2(600,1350)}^1$ $15P_{3(750,1250)}^1$ corresponding to (d-a) is stored on the magnetic drum. The vowel segment $17P_a^8$ is resident in the core memory.

A particular segment for the silent interval is also resident in the core memory. Its duration and amplitude parameter (a) are set at about 200 msec and 0 respectively.

The number of dyads is about 180 and the necessary memory space for storing these data amounts up to about 170 kbits. The effects on performance of the system, in case of using dyad segment sequences as the actual set of compilation elements, will be discussed later in Section 2.7.

2.3.2 System Configuration

Global configuration of the proto-type of the on-line, real-time, multiple speech output system is shown in Fig.2-3. It is made up of two main components. One is a computer, which receives input sentences*, processes them and then presents output in a numerical form of zero-crossing waves successively in on-line, real-time mode. The other is a synthesizer part where each synthesizer is allocated to each active user and independently produces speech sounds. These components mutually communicate through a peripheral adapter (general purpose interface for peripherals). Fig.2-4 shows the general view of the synthesizer part containing two synthesizers experimentally constructed in hardware.
Data transfer rate through the coaxial cable is so high that the synthesizer part must be necessarily installed near the computer for the reliable data transmission.

A medium size computer (NEAC 2200/200, core memory 32 kchs, cycle time 2 μsec, data channel transfer rate 166 kchs(6 bits)/sec) is used, to which a wide variety of peripheral units can be connected through the peripheral adapter designed for more convenient use in on-line, real-time applications.

Computer programs can be run either in a normal or an interrupt mode. Because of one level interruption facility of the hardware of the computer, interrupt signals can be accepted immediately at their occurrence, only when the computer is operated in the normal mode.

* see p.44

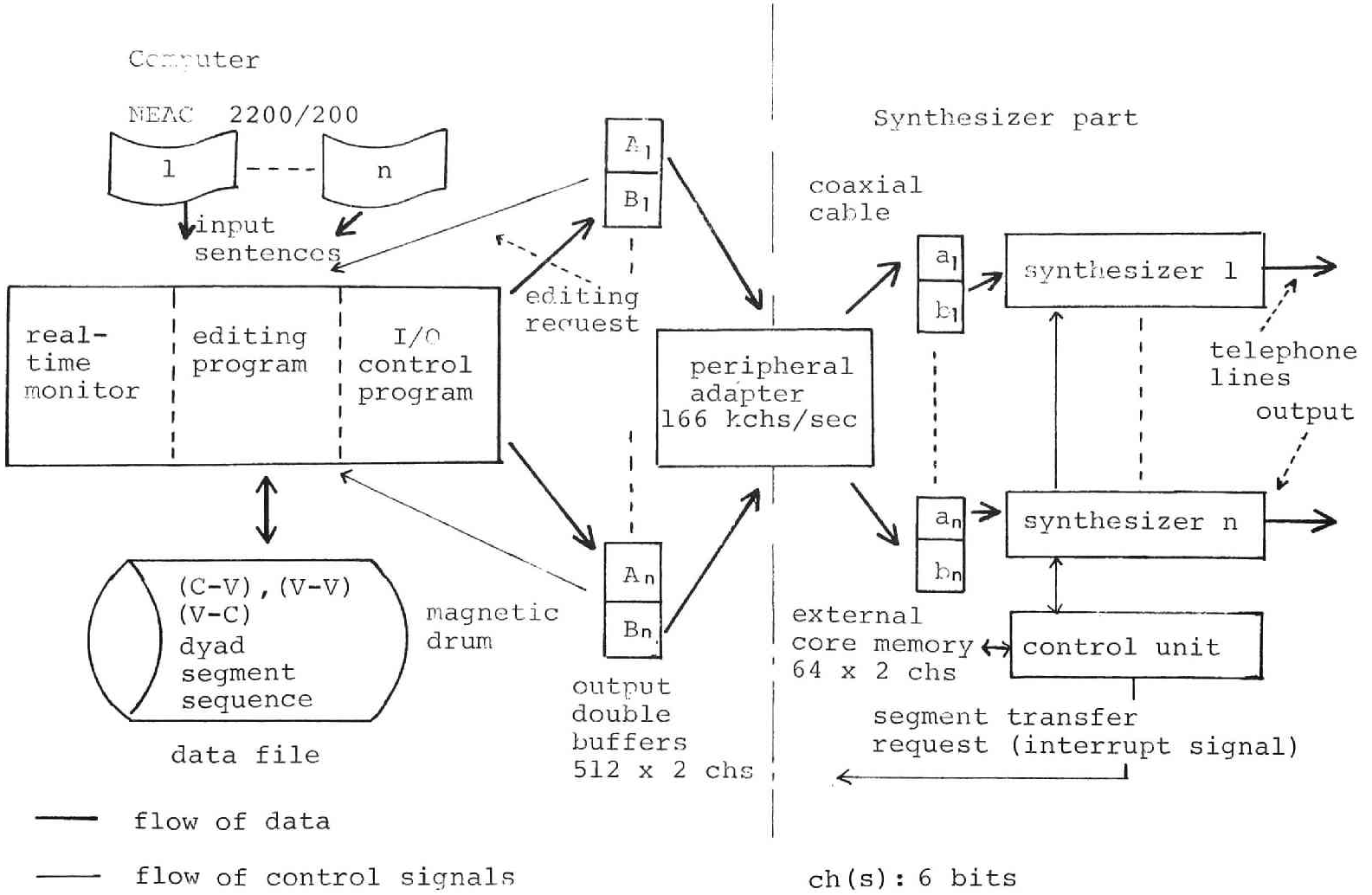


Fig. 2-3 Global configuration of the system

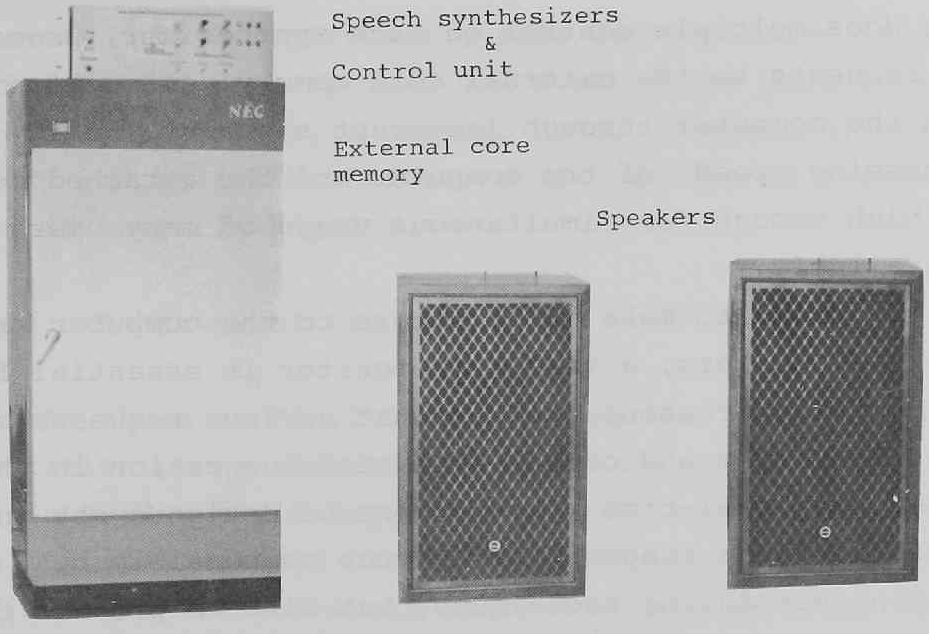


Fig. 2-4 Synthesizer part

2.3.3 Outline of the System Behaviors

The main function of an editing program in Fig.2-3, is to assemble up a dyad segment sequence in a complete form in one of the output double buffers (512 x 2 chs) allocated to each active user. It includes retrieval of a dyad segment sequence, data transfer of part of it from the magnetic drum (data file), and rearrangement of it in a complete form.

In response to a segment transfer request from the synthesizer part, each segment of dyad segment sequence rearranged in the output double buffers, is transmitted via the coaxial cable, and stored in one of the areas of an external core memory (64 x 2 chs for each user). Speech sound requested to be output, is converted at a synthesizer, and transmitted over a telephone line to the user. The control unit equipped in the synthesizer

part does multiple control on each synthesizer, processes R/W requests to the external core memory, and communicates with the computer through interrupt signals. The processing speed of the computer and the attached devices is high enough for simultaneous usage of many users to be available.

In order to make multi-access to the computer from many synthesizers, a real-time monitor is essential for the following reasons. One is that various requests may occur at random and cause unexpected congestion in the system. The real-time monitor must be designed not only to reduce these congestions as much as possible by applying scheduling techniques, but also to process them without any interference of each other. The other reason is that simple multi-programming technique enables the central processor to spend a great proportion of its time efficiently. In the experimental system, both speech output and background programs are supposed to be run. The program for speech output is allotted a higher priority with pre-emption and processed in on-line, real-time mode. While, background programs can be run with a lower priority only at the idle times and I/O waiting periods of the real-time processings. The real-time monitor executes these functions related to task switching successfully.

2.4 Hardware Configuration and its Behavior

In this section, the hardware configuration and its

behavior in the synthesizer part are presented in somewhat detailed description. The synthesizer part contains speech synthesizers, a control unit, an external core memory and so forth. General configuration of the hardware is shown in Fig.2-5.

2.4.1 Behavior of the Hardware

As shown in Fig.2-5, one of the OXI data (OXI_j), its repetition parameter (r) and amplitude parameter (a) are set in P-, R-, and A-registers of the synthesizer. Under the multiple control of the control unit, OXI_j set in P-register is reduced by one at every clock pulse of 50 μsec (sampling) pulse generator. When the content of the P-register becomes zero, the state of the T-FF is reversed, and at the same time OXI data-request-signal is emitted and the next digit OXI_{j+1} is read into P-register from $a_i(b_i)$ by the control unit. Here, $a_i(b_i)$ means one of the areas of the external core memory allocated to the user (i).
The content of R-register is reduced by one at the time that the end marker (*) of the segment in $a_i(b_i)$ is detected. If it is not zero, the OXI sequence in $a_i(b_i)$ is transmitted repeatedly to P-register from its beginning. During this operation, an amplitude control circuit keeps the amplitude of the resulting zero-crossing wave at the same level as indicated in the content of A-register.

If the content of R-register becomes zero, the OXI sequence, repetition parameter (r), and amplitude parameter (a) in $b_i(a_i)$ are read out and stored into P-, R-, and A-registers respectively and at the same time an interrupt signal is transmitted to the computer requesting

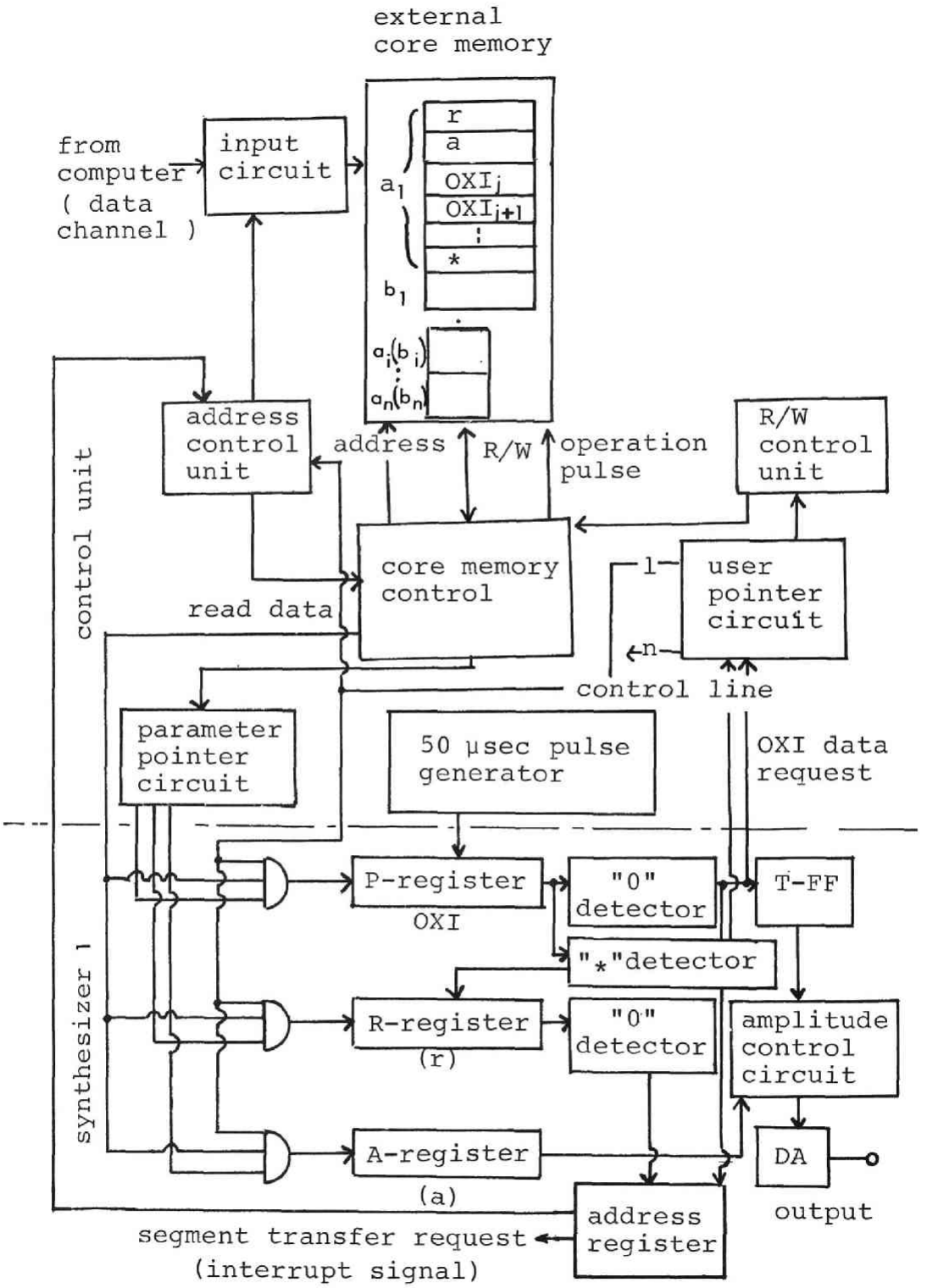


Fig. 2-5 Hardware configuration of the system

that the next segment be transferred into $a_i(b_i)$.

By repeating the same operations, speech sounds such as Fig.2-2 are produced through each synthesizer and are transmitted to the user via telephone line. The envelope of the waves is modified in a natural form by several kinds of filters in DA block.

2.4.2 Configuration of the Speech Synthesizer

Each synthesizer allocated to an active user, is composed of P-,R-, and A-registers, an address register, various signal detectors, and an amplitude control circuit.

In P-,R- and A-registers are set the corresponding parameters through circuits gated by signals supplied by the control unit. Each register is constructed of add-one counter that can preset arbitrary numbers. Many detectors transmit signals to the control unit to indicate their statuses and require the necessary processings.

The content of an address register of each active user is added by one, when the content of P-register comes to zero. It returns to indicate the 3rd relative location in each area of the external core memory when the end marker (*) is detected. When the content of R-register becomes zero, it goes back to the first.

The amplitude control circuit is designed to amplify the resulting zero-crossing wave up to the level indicated by an amplitude parameter (a). The wiring diagram is shown in Fig.2-6. Digital data quantized into 4 bits and stored in A-register, are converted to an analog voltage by the operational amplifiers. This is a power supply voltage for switching circuit. Zero-crossing waves from T-FF in

Fig.2-5 are input to the switching circuit, the output of which is modified in its amplitude.

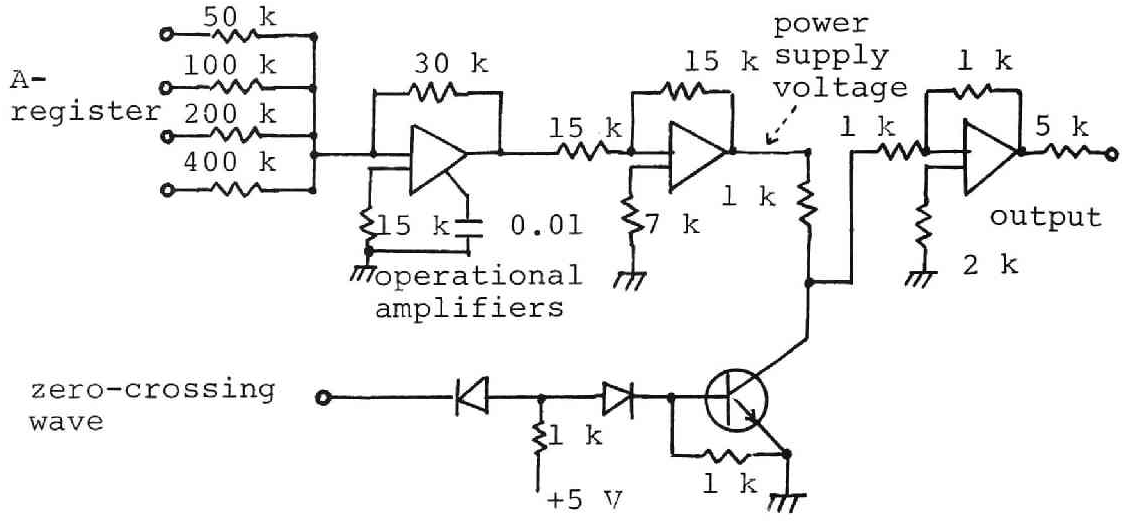


Fig. 2-6 Amplitude control circuit

2.4.3 Configuration of the Control Unit

The control unit supervises each synthesizer under multiple control and enables it to produce speech sounds independently. A notable characteristic of the control unit is that it is an asyncronous piece of equipment. In other words, the control unit uses its time to process various requests randomly occurring from each synthesizer, only at the periods of their occurrence. It allocates residual time , during which no requests appear, to other processings. On the other hand, another controlling method that could be easily conceived of, is to control each synthesizer syncronously so as not to cause any contention. It would spend allotted amount of its time to poll

and process each request in a cyclic way.
Comparing with this syncronous controlling method ,one of the benefits of the asyncronous controlling is to enable an efficient use of the processing times. For example, the external core memory can be used to write data through the data channel throughout almost all the time except when it is occupied by sparse requests for reading data to each synthesizer, as shown in Fig.2-8. It results in some reduction of holding time of the data channel. But, one of the disadvantages is that it is somewhat complex in its operating mechanism. This is mainly due to additional circuits in order to prevent some serious timing conditions that must be taken into account generally in making asyncronous systems. Another disadvantage is that requests may be made to wait in too long a queue for the control unit to process them within a certain limited time. This is mainly due to the random arrivals of requests and probably lets the system result in fatal confusion.

An elaborate operation should be performed in order to prevent this fatal condition induced by the rare occurrence of the delay of the real-time processings.

Upper half of Fig.2-5 shows a block diagram of the control unit, which is composed of an input circuit, a user pointer circuit, a core memory R/W control unit, a parameter pointer circuit and a 50 μsec pulse generator. These are described in detail as follows.

(1) Input Circuit

The computer accepts a segment transfer request in an interrupt mode and then transmits a requested one into

one of the areas of the external core memory allocated to each active user through the data channel. The input circuit is an interface between the data channel controller and the external core memory. The external core memory is in write mode throughout almost all the time, in order to receive data at the same rate as the data channel (166 kchs/sec). The end of transmission is detected by reception of an end marker (*) and notified to the data channel controller by a status line.

(2) 50 μsec Pulse Generator

At every 50 μsec pulse supplied by this circuit,the content of P-register of each synthesizer is reduced by one. When it becomes zero, the next digit of OXI stored in the external core memory must be read out within the next 50 μsec time interval. If not, the pulse width of the produced zero-crossing wave could not be equal to its designed value. In the worst case, however, core memory access must be done four times to read the end marker (*), repetition parameter (r), amplitude parameter (a) and the first OXI, when the content of R-register becomes zero. It is a serious problem in the experimental system, that this processing requires more than 50 μsec. But for the effective operation to solve this problem, it would be impossible to produce multiple speech outputs. However, the probability is quite rare that these worst cases would occur simultaneously among many synthesizers. In order to minimize bad effects by this delay of the real-time processing, one method is to forbid the next pulse to reduce the content of each P-register when one of the R-registers is zero. Although the zero-crossing interval of the produced speech is prolonged at least

50 μsec more than it should be, the resulting speech output is gracefully degraded and is hardly influenced for human understanding.

(3) User Pointer Circuit

This circuit has a very simple loop structure as shown in Fig.2-7, and plays an essential role in multiple controlling.

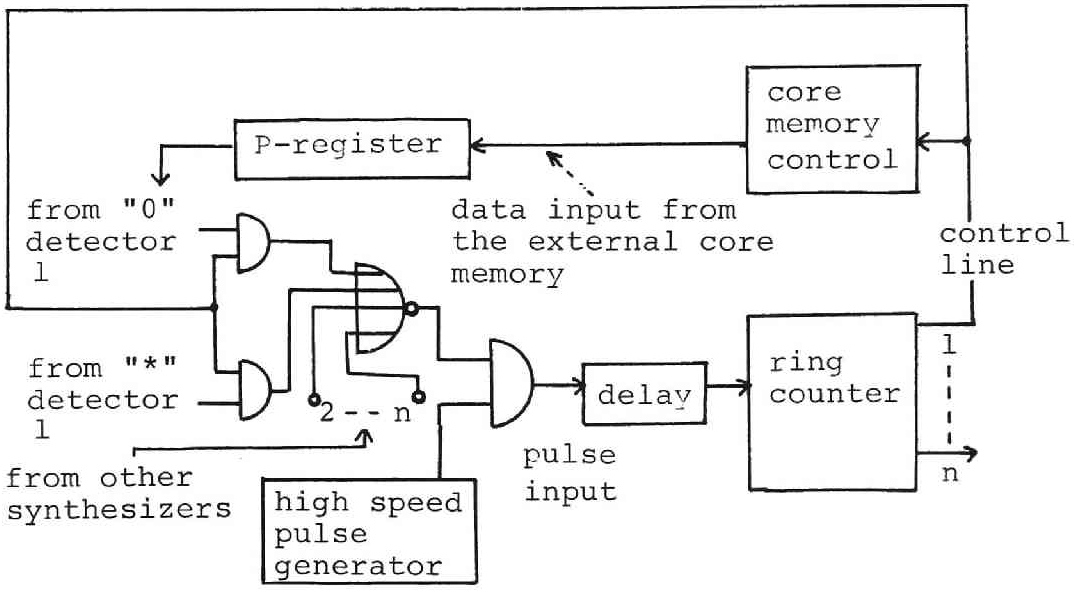


Fig. 2-7 Block diagram of user pointer circuit

Pulse string emitted from the generator is input at high frequency, e.g.2 MHz, to the ring counter, one element of which can successively make the connected control line exhibit one-condition. When the active user pointed by this control line requests the processings, the pulse input is inhibited from entering the ring counter. That control line is held at one-condition, until it has finished processings. After the processings (e.g., data

input from the external core memory) are finished in uncertain time interval, inhibition of the pulse input is automatically released and the next control line is activated. Fig.2-8 shows one of the timing conditions under multiple control of the system.

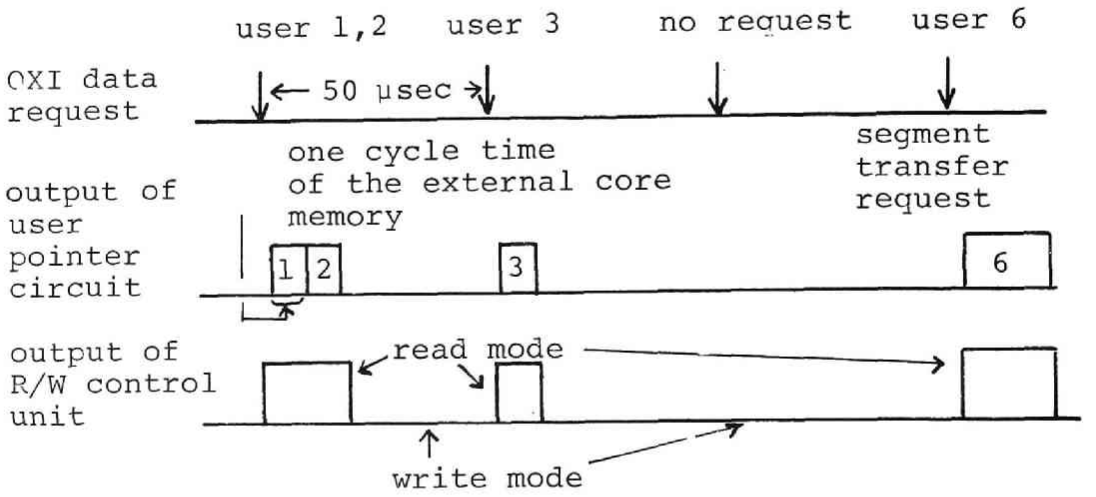


Fig. 2-8 Some timing condition under multiple control

(4) Parameter Pointer Circuit

This circuit is used to select one of the P-, R- and A-registers and to store data when each parameter is read out from the external core memory.

(5) Address Control Unit

This circuit controls read and write (R/W) operations on the external core memory. The core memory has a capacity of 1024 chs, 64 x 2 chs of which are allocated to each user. It is addressable by 10 bits, the more significant 3 bits of which are used to identify each

active user. The seventh bit indicates the R/W area of the double buffers of the external core memory.
In read mode, the address control unit indicates the content of the address register of the active user which is pointed by the user pointer circuit. On the other hand, in write mode, it indicates writing area of 10 bits, the more significant 3 bits of which are sent from the computer to identify the active user before segment transfer.

(6) R/W Control Unit

This also controls R/W operations on the external core memory. Read requests are allotted higher priority for access to the core memory than write requests, because the formers must need more severe real-time processings. Read requests interrupt segment transferring and are processed with pre-emption. Of course, this mode switching is performed without missing any data.

2.5 Software Configuration and its Behavior

Software for producing the speech outputs is mainly composed of three sub-programs. These are an editing program, a real-time monitor and an I/O control program. The following are detailed descriptions of these programs.

2.5.1 The Real-time Monitor and I/O Control Program

Fig.2-9 shows the outline of the flowchart of the software. An interrupt signal occurs to the computer requesting the next segment to be transferred, when one of the double buffers $a_i(b_i)$ of the external core memory becomes empty. If the system is operated in the normal mode, the real-time monitor checks the interrupt source and transfers control to the I/O control program. Otherwise, requests are made to wait until the system resumes the normal mode.

The I/O control program transmits a required segment in $A_i(B_i)$ to the area $a_i(b_i)$ of the external core memory. Here, $A_i(B_i)$ is one of the output double buffers allocated to the user (i). The required segment is selected by referring to a table listing relative addresses of each segment in the output double buffers. This table is prepared during the operation of the editing program.

By repeating these operations, the segments in $A_i(B_i)$ are fully transmitted. Then, a segment in $B_i(A_i)$ begins to be output and an editing request to store dyad segment sequence into $A_i(B_i)$ is formed in a queue and made to wait until the editing program finishes to serve other preceding queues. After I/O control program completes these tasks, the system resumes the normal

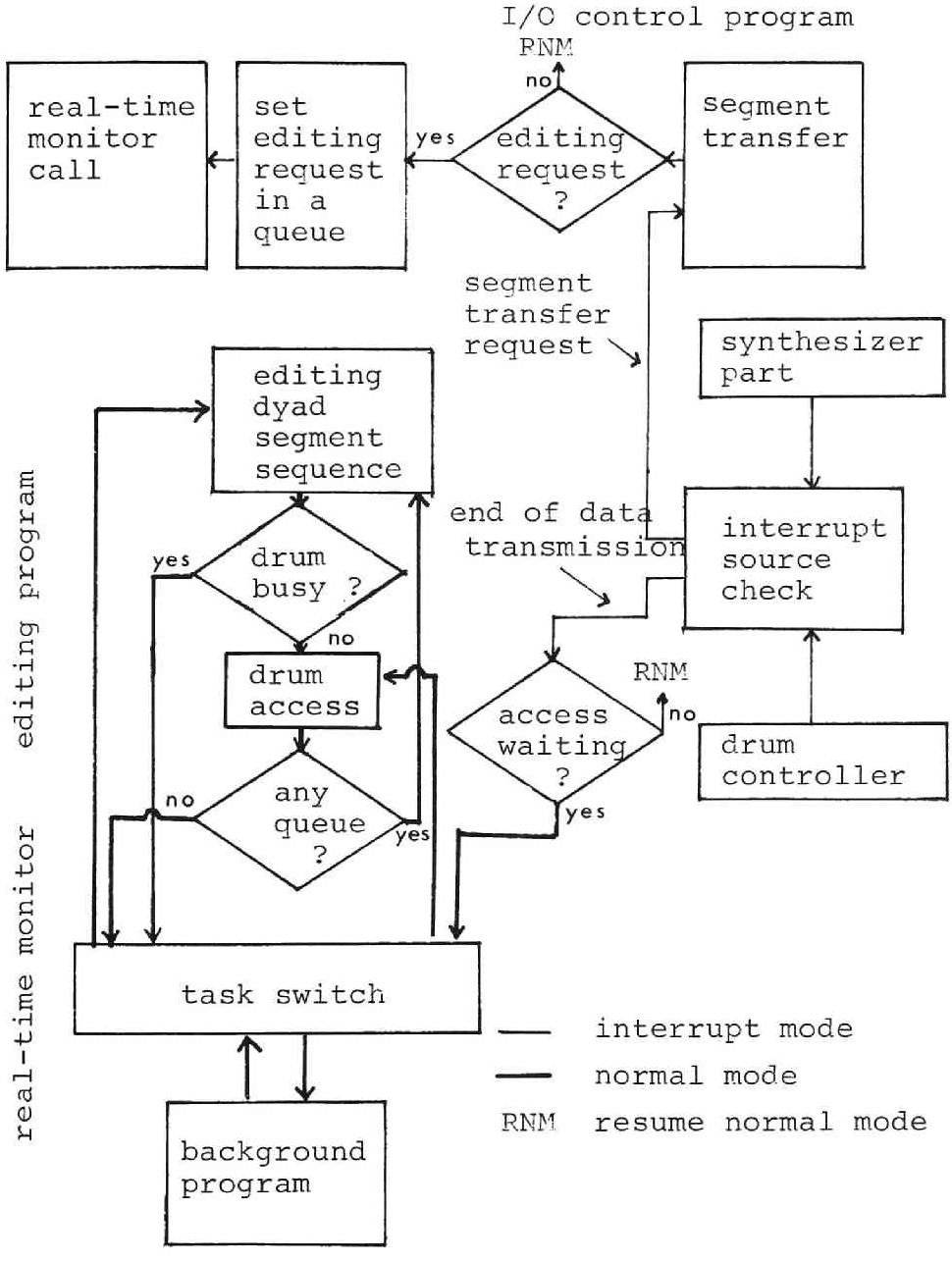


Fig. 2-9 Flowchart of the software

mode again.

The editing program contains I/O operation on the magnetic drum. If a controller of the drum is busy with data transmission of a previous request, then the real-time monitor makes task switching to a background program to process another task of lower priority.
An interrupt signal from the drum controller notifies the end of data transmission to CPU. Then the real-time monitor makes task switching from the background program to the editing program for the next data transmission from the magnetic drum. If the processing of the editing program is finished, the real-time monitor checks the content of the queue for the editing request. In case of no queue in line, the real-time monitor switches its control from the editing program to the background. Otherwise, the editing program continues to process the next queue.

2.5.2 The Editing Program

The main functions of the editing program include; (1) pick up a request formed in the queue; (2) search for a drum address location of an output dyad segment sequence; (3) prepare the table for I/O control program; (4) transfer data from the magnetic drum (drum access) and (5) rearrange dyad segment sequence in a complete form.

Table 2-2 lists characters available for use in input sentences of the system, which are punched in Roman letters, and fed into the computer for speech output.

Table 2-2

Characters available for use

A	T	M	H
I	K	D	Z
U	S	Y	P
E	N	G	Space
O	R	W	
-	long vowel		
.	period		

In the context of vowel-consonant (or vowel), a vowel part is arranged in one of the output double buffers before the glide part is transferred from the magnetic drum. In case of vowel-space or syllabic nasal-space contexts, a particular segment whose amplitude parameter and duration are zero and 200 msec respectively, is stored in the output double buffers, being preceded by either vowel or syllabic nasal segments.

In the context of nasal-vowel, the segment of a nasalized vowel must be selected in the next coming vowel-consonant (or space) context. In the context involving /w/ or /y/, it is temporarily dealt with as a vowel /u/ or /i/, but the segment of the vowel part is eliminated at the phase of rearrangement of data. If a long vowel appears, all the processing has to do is to give a longer value to the repetition parameter of the vowel segment.
It may be clearly expressed by an additional symbol whether a letter n is either a nasal or a syllabic nasal, where both of them can't clearly be identified in such a context as a Japanese word "mini"(ミニ, 民意).

All of these operations are presented in the flow-chart of Fig.2-10.

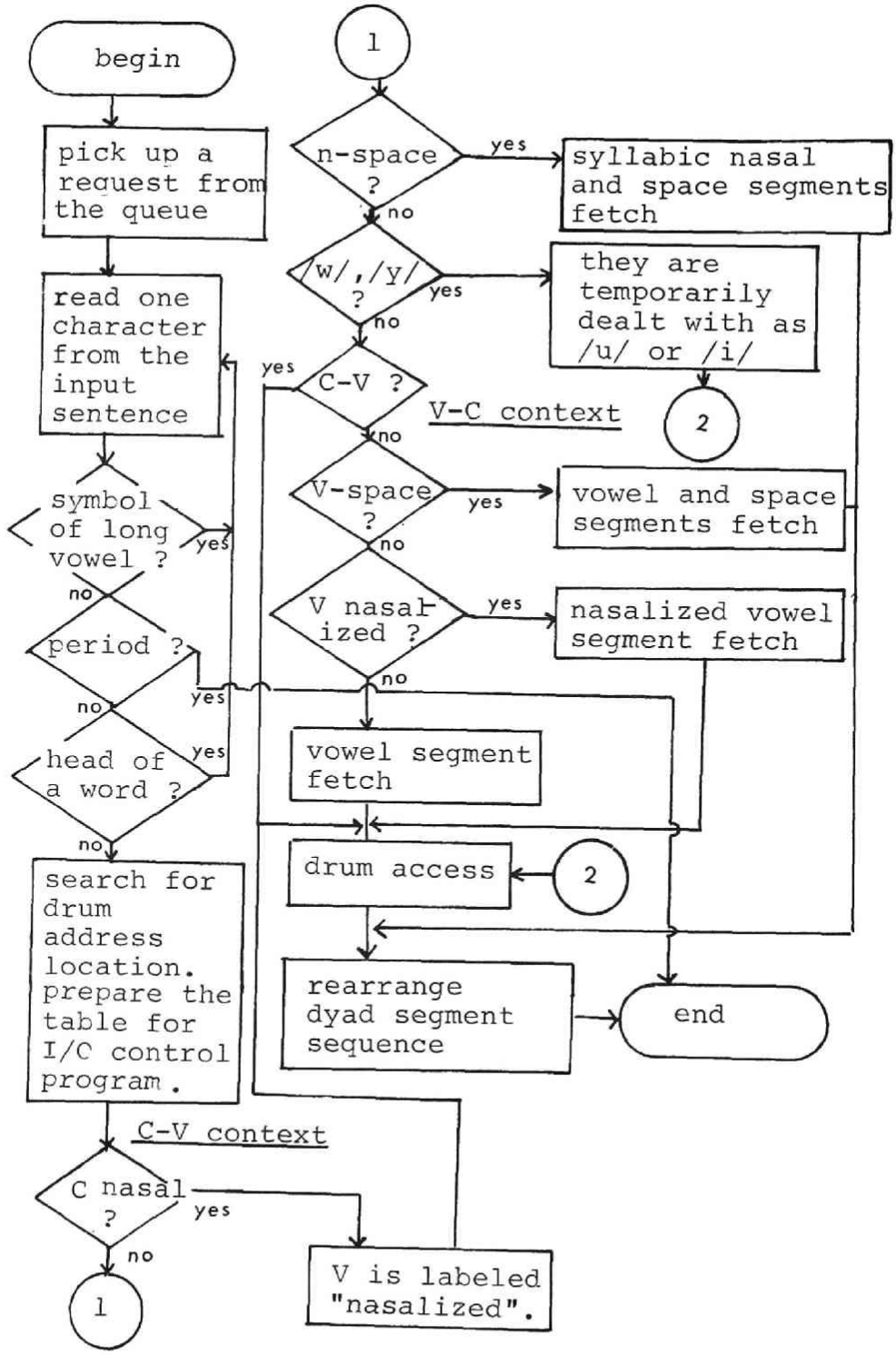


Fig. 2-10 Flowchart of the editing program

2.6 Speech Synthesizer Controlled by a Minicomputer

Recently, minicomputers have been popularly connected to larger computers for controlling a wide variety of peripheral units. They are very high in speed and inexpensive. They can perform the same operations as in the synthesizer part in Fig.2-3. Possible usages of a minicomputer replacing the hardware logic of the synthesizer part would be as follows:

(1) Partial replacement of the hardware logic by software

Fig.2-11 shows one of the configurations. In this, all of the hardware except P-,and A-registers and DA block in Fig.2-5 are replaced by the software of the minicomputer (HITAC 10-II).

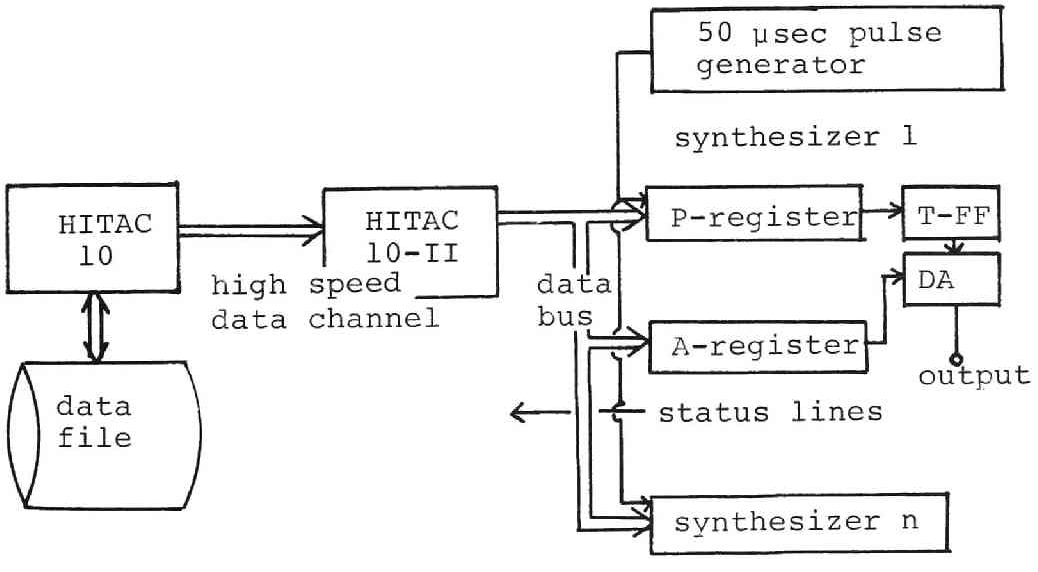


Fig. 2-11 Speech synthesizer controlled by a minicomputer (1)

OXI data are successively transmitted to each P-register through a data bus. Data transferring is through an accumulator of the minicomputer (i.e., programmed data transfer). A control program always supervises through status lines whether each P-register requires data transfer of one digit of OXI. If a request is sensed, one digit of OXI is immediately read out and sent to the P-register. This processing must be finished in 50 μsec. Unfortunately, it takes more than 20 μsec when the minicomputer HITAC 10-II (cycle time; 0.9 μsec) is used. Then, a number of multiplicity (simultaneous outputs) is limited to one or two. However, attached hardware devices in Fig.2-11 are extremely simple in logic and constructed of about 10 integrated circuits.

(2) Complete replacement of the hadware logic by software

The configuration of this type is shown in Fig.2-12. A minicomputer (Melcom 70) receives a segment successively sent from NEAC 2200/200. A program of the minicomputer converts a numerically expressed OXI data in a segment into the data string. The length and absolute value of each data in the string are equal to the contents of one digit of OXI data and the amplitude parameter (a) of the segment, respectively. The data string for the next digit of OXI is reversed in its sign.

Each data contained in the string is transmitted at every sample pulse (50 μsec) to a D/A converter. For multiple speech outputs, a high speed multiplexor channel must be installed in the minicomputer.

This scheme is now beirg examined by Ohtani et al.[35]

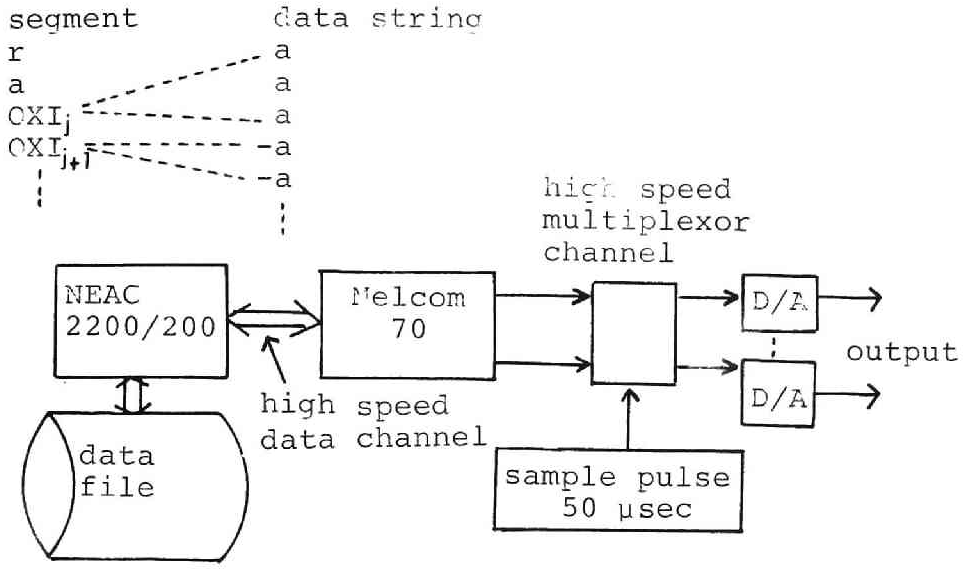


Fig. 2-12 Speech synthesizer controlled by a minicomputer (2)

(3) Use of a minicomputer only as the external core memory

This configuration is quite similar to that of the proposed system in Fig.2-3 except that some functions of the control unit would be replaced by software. However, a high speed multiplexor channel with rather complicated interface to the synthesizer part, would be required in order to realize multiple speech outputs.

2.7 Evaluation of the System

In designing a real-time system, many important problems must be successfully solved. First, it is very essential to know how severe the required real-time is for processings of random arrival requests. Satisfactory specification of the system strongly depends on this consideration. Second, it is very important to consider the means to prevent the system from being confused by unexpected delay of real-time operations and/or some severe timing conditions.

Evaluating the system must be considered totally in CPU spent time, data channel spent time, amount of used memory and serviceability. The first step for evaluating the system performance is to get statistical data. An analysis of these data often presents system's bottleneck points and shows the way for improving system performance.

2.7.1 Arrival Distribution of the Requests

Arrival distributions of both editing and segment transfer requests are described. Henceafter, a numbered element in the set of dyad segment sequence is abbreviated as D(k). P(k) means the frequency in use of D(k) in the normal contexts of Japanese sentences, and τ(k) corresponds to the actual duration of D(k) when it is produced in speech sounds. τ(k) is equal to an arrival interval of an editing request, because it occurs just at the end of speech sounds of D(k). Fig.2-13 shows the arrival distribution of the editing request from an active user. A value of Y-axis of the graph is a sum of P(k) with

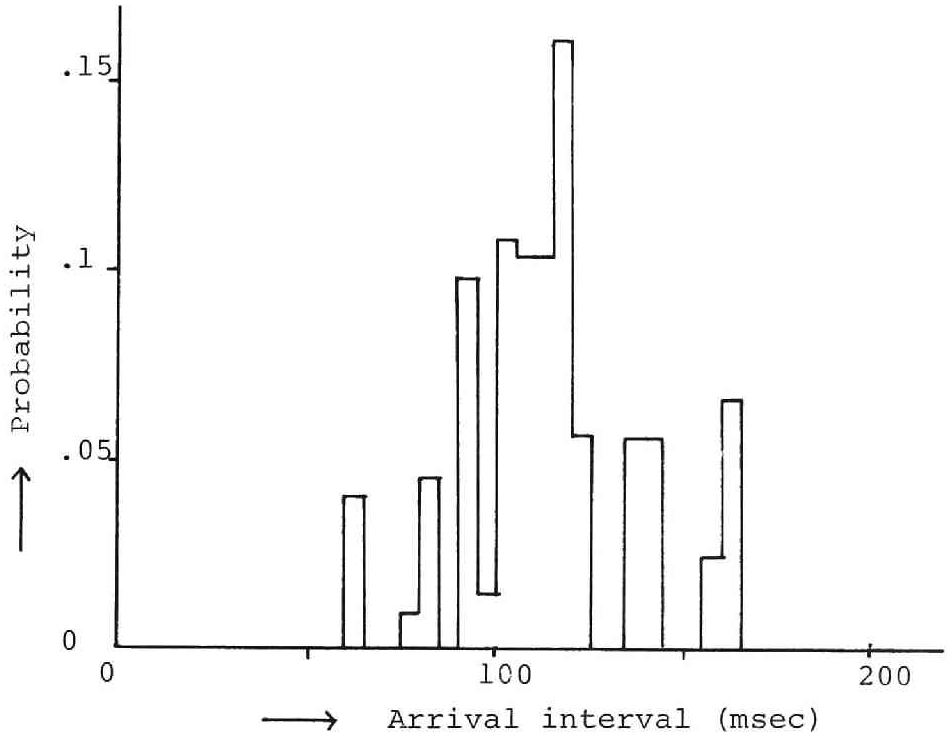


Fig. 2-13 Arrival distribution of editing requests

almost equal $\tau(k)$ (X-axis).

The mean arrival interval of the editing request T_F is expressed by,

$$T_F = \sum_k P(k)\,\tau(k) \qquad \text{(sec)}. \tag{2.1}$$

Calculation which resulted from a list of statistical data on D(k) shown in Table 2-3,[44] indicates that T_F is nearly equal to 110 msec.

On the other hand, the mean arrival interval of the segment transfer request is equal to the mean duration of the segment. M(k) is denoted as a number of segments contained in D(k). The mean arrival interval of the segment transfer request is expressed by,

$$T_S = \sum_k P(k)\tau(k)/M(k) \qquad \text{(sec)}. \tag{2.2}$$

It results in about 22 msec.

Table 2-3

Statistical data of dyad segment sequence D(k)

	P(k)	τ(k)	X(k)	H(k)	M(k)		P(k)	τ(k)	X(k)	H(k)	M(k)
ta	0.020	116	215	144	7	to	0.026	116	198	146	7
ti	0.007	101	365	313	7	tu	0.009	105	316	264	7
te	0.016	116	259	207	7	ty	0.002	101	365	313	7
ka	0.022	161	286	214	7	ko	0.015	161	261	189	8
ki	0.010	161	392	320	8	ku	0.014	161	264	192	8
ke	0.006	161	356	284	8	ky	0.001	161	392	320	8
sa	0.006	108	1079	277	7	so	0.007	108	1060	256	7
si	0.020	158	1153	351	7	su	0.016	108	1065	263	7
se	0.006	108	1117	315	7	sy	0.006	155	1153	351	7
na	0.016	91	235	107	7	no	0.024	91	211	83	5
nu	0.001	91	322	97	5	ne	0.002	91	283	155	5
ny	0.000	91	323	195	5	ni	0.015	91	323	195	5
ra	0.010	82	185	107	5	ro	0.004	82	162	84	5
ri	0.008	82	279	88	5	ru	0.013	82	166	88	5
re	0.009	82	231	153	5	ry	0.001	82	279	88	5
ma	0.014	90	144	74	4	mo	0.011	90	127	57	4
mi	0.004	90	242	172	4	mu	0.001	90	136	66	4
me	0.003	90	188	118	4	da	0.008	105	223	150	6
do	0.006	105	211	138	6	de	0.016	105	262	189	6
ya	0.005	63	207	221	7	yo	0.014	63	175	189	7
yu	0.008	63	189	200	7	ga	0.016	117	239	201	7
go	0.002	116	211	173	7	gi	0.001	116	345	307	7
gu	0.001	116	239	201	7	ge	0.002	116	321	283	7
gy	0.000	116	345	307	7	ha	0.004	104	284	194	6
ho	0.003	122	256	181	6	hi	0.004	118	774	382	8
hu	0.002	124	272	182	6	he	0.001	84	346	256	6
hy	0.001	117	774	382	8	za	0.001	129	721	385	11
zo	0.001	129	693	357	11	zi	0.004	147	793	461	11
zu	0.004	129	699	361	11	ze	0.001	129	755	417	11
zy	0.004	147	797	461	11	ba	0.004	77	129	105	4
bo	0.001	77	119	95	4	bi	0.001	77	175	151	4
bu	0.003	77	120	96	4	be	0.001	77	155	131	4
by	0.000	77	175	151	4	pa	0.001	107	175	119	5
po	0.000	107	167	111	5	pi	0.000	107	228	171	5
pu	0.001	107	171	115	5	pe	0.000	107	203	147	5
wa	0.017	63	113	127	7	at	0.014	113	312	134	4
ak	0.019	122	301	125	5	as	0.018	113	280	102	4
an	0.012	122	301	125	5	ar	0.018	122	301	145	5
am	0.006	113	274	96	4	ad	0.005	113	282	104	4
ay	0.000	140	403	272	7	ag	0.004	122	313	137	5

Table 2-3

(Continued)

	P(k)	τ(k)	X(k)	H(k)	M(k)		P(k)	τ(k)	X(k)	H(k)	M(k)
aw	0.003	140	337	167	8	ah	0.003	122	301	125	5
az	0.003	122	305	129	5	ab	0.001	95	255	75	3
ap	0.000	95	255	75	3	aa	0.003	140	350	180	8
ao	0.003	140	329	159	8	ai	0.015	140	442	272	8
au	0.001	140	338	168	8	ae	0.003	140	398	228	8
an	0.010	140	331	161	8	a	0.010	326	430	43	2
ot	0.012	106	204	90	4	ok	0.015	115	214	104	5
os	0.008	106	191	79	4	on	0.012	115	211	101	5
or	0.008	115	214	104	5	om	0.007	106	195	83	4
od	0.001	106	195	82	4	oy	0.005	142	347	196	8
og	0.003	115	225	112	5	ow	0.004	142	220	116	8
oh	0.004	115	214	94	5	oz	0.002	106	189	97	4
ob	0.002	97	180	66	3	op	0.000	97	176	53	3
oa	0.002	142	253	149	8	oo	0.004	142	238	134	8
oi	0.004	142	345	194	8	ou	0.001	142	207	116	8
oe	0.002	142	310	206	8	on	0.004	142	233	129	8
o	0.001	330	350	33	2	it	0.022	104	506	184	4
ik	0.011	113	565	245	5	is	0.007	104	502	180	4
in	0.012	113	537	217	5	ir	0.007	113	545	225	5
im	0.010	104	496	174	4	id	0.005	104	499	177	4
iy	0.002	140	700	386	8	ig	0.005	113	565	245	5
iw	0.004	140	571	269	8	ih	0.002	113	553	233	5
iz	0.003	113	525	205	5	ib	0.002	95	463	139	3
ip	0.000	95	463	139	3	ia	0.001	140	593	278	8
io	0.003	140	557	243	8	ii	0.003	140	700	386	8
iu	0.000	140	573	269	8	ie	0.001	140	660	356	8
in	0.004	140	558	244	8	i	0.001	321	600	67	2
ut	0.008	109	208	90	4	uk	0.008	118	219	103	5
us	0.006	109	202	84	4	un	0.008	118	229	113	5
ur	0.006	118	227	111	5	um	0.004	109	200	82	4
ud	0.003	109	204	86	4	uy	0.002	145	358	251	8
ug	0.003	118	254	140	6	uw	0.002	145	259	149	8
uh	0.001	118	221	105	5	uz	0.001	118	221	105	5
ub	0.001	98	185	65	3	up	0.000	98	181	61	3
ua	0.006	145	290	149	8	uo	0.001	145	230	120	8
ui	0.003	145	366	256	8	uu	0.000	145	243	149	8
ue	0.001	145	331	221	8	un	0.002	145	245	152	8
u	0.007	332	460	33	2	et	0.008	102	358	144	4
ek	0.006	111	407	156	5	es	0.010	102	354	140	4
en	0.004	111	387	175	5	er	0.005	111	391	179	5

Table 2-3

(Continued)

	P(k)	τ(k)	X(k)	H(k)	M(k)		P(k)	τ(k)	X(k)	H(k)	M(k)
em	0.004	102	348	134	4	ed	0.004	102	356	142	4
ey	0.001	138	539	333	8	eg	0.001	111	407	195	5
ew	0.003	138	443	237	8	eh	0.001	111	389	177	5
ez	0.001	111	382	170	5	eb	0.001	93	321	105	3
ep	0.000	93	321	105	3	ea	0.001	138	414	208	8
eo	0.002	138	385	179	8	ei	0.010	138	539	333	8
eu	0.000	138	443	237	8	ee	0.000	138	370	312	8
en	0.007	138	393	197	8	e	0.001	325	500	49	2
nt	0.002	90	146	76	4	nk	0.002	99	159	91	5
ns	0.003	90	138	68	5	nn	0.005	99	159	91	5
nr	0.001	99	156	88	5	nm	0.001	90	141	71	4
nd	0.005	90	140	70	4	ny	0.000	126	296	234	8
ng	0.003	99	158	90	5	nw	0.001	126	203	143	8
nh	0.000	99	159	91	5	nz	0.002	99	157	89	5
nb	0.000	81	121	49	3	np	0.001	81	166	100	6
n	0.001	313	310	25	2						

2.7.2 Information Amount to be Controlled

H(k) is a necessary memory capacity to store D(k). The mean amount of controlling information K_1 can be calculated by the following equation,

$$K_1 = \sum_k P(k)H(k)/\tau(k) \quad \text{(chs/sec)}. \qquad (2.3)$$

It results in about 1.5 kchs/sec. Fig.2-14 shows the distribution of the data length of the dyad segment sequence.

X(k) is a number of zero-crossings in produced speech sounds of D(k). The mean number of zero-crossings in one second K_2 can be calculated by,

$$K_2 = \sum_k P(k)X(k)/\tau(k), \qquad (2.4)$$

The result of this is 3.05×10^3 . Then, the repetition

parameter makes the controlling information reduced by

$$K_1/K_2 \fallingdotseq 1/2 \quad . \tag{2.5}$$

The mean length of the dyad segment sequence L_F, and that of the segment L_S are both expressed by the equations.

$$L_F = \sum_k P(k)H(k) \qquad \text{(chs)} \tag{2.6}$$

$$L_S = \sum_k P(k)H(k)/M(k) \qquad \text{(chs)} \tag{2.7}$$

Calculation shows that L_F is about 140 chs and L_S is about 28 chs respectively.

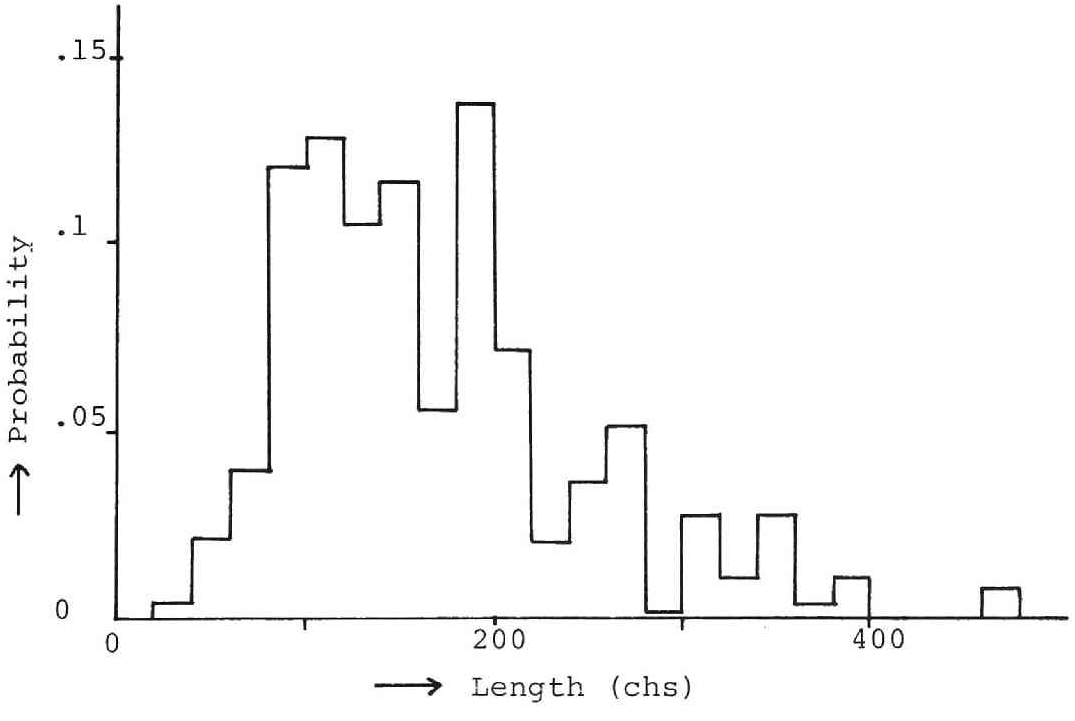


Fig. 2-14 Distribution of the data length of the dyad segment sequence

2.7.3 Number of Multiplicity of the System

The number of multiplicity n is dependent on the following factors:

(1) Characteristics of the control unit

The content of each P-register is reduced by one at every 50 μsec. The OXI data request must be processed within the next 50 μsec time interval. The following inequality must be employed in the worst case where all the synthesizers request for transferring OXI data.

$$4t_{ac}n+T_{R/W}\leqq 5 \times 10^{-5} \qquad (2.8)$$

where t_{ac} is a cycle time of the external core memory and $T_{R/W}$ is set-up-time for the R/W control unit in the synthesizer part. Core memories with fast cycle time have been developed recently and $T_{R/W}$ is less than 6 μsec. Then, maximum number of multiplicity of the system is not so heavily dependent on this inequality.

(2) Speed of the data channel

Ratio of memory cycles used by the data channel controller to the total effective time of the channel e_1 is expressed by,

$$e_1=K_1/C \qquad (2.9)$$

where K_1 is the mean amount of controlling information and C is the transfer rate of the data channel (166 kchs/sec). Calculation shows e_1 is nearly equal to 0.8 %. On the other hand, the ratio of read mode in the external core memory in one second e_2, is expressed for each synthesizer,

$$e_2 = t_{ac} K_2 \qquad (2.10)$$

where K_2 is the mean number of zero-crossings in one second. Calculation shows e_2 is about 2 %.

Since simultaneous read/write operations on the external core memory is impossible, the following inequality holds for the data channel between the computer and the synthesizer part.

$$nK_1 \leqslant C(1-ne_2) \qquad (2.11)$$

Right hand side of Eq.(2.11) means the available transfer rate of the data channel. Calculation results that n is less than 35. The transfer rate of the data channel does not severely influence the maximum number of multiplicity of the system.

Holding time of the data channel in one second is,

$$e_1/(1-ne_2). \qquad (2.12)$$

This is a little larger than e_1 , because read requests interrupt the data transfer through the data channel with pre-emptive priority.

(3) Waiting time for segment transferring

A segment transfer request must be processed before the buffers become empty. In the worst case when simultaneous requests occur from all synthesizers, the following inequality holds,

$$n(\max(W_t/(1-(n+1)e_1/N), L_S/C(1-ne_2)) \leqslant T_S . \qquad (2.13)$$

Here, W_t means the necessary operation time for an I/O control program. The first term in left hand side of Eq.(2.13) is the effective operation time for the I/O

control program. The second means necessary time for transferring a segment (its mean length L_S). Calculation shows $\underline{n}$ is less than 40 . This factor is not so important in the performance of the system.

(4) CPU spent time

This is a rate of the time in which CPU is occupied by the speech production to the total effective time. It is,

$$H_t/T_F + W_t/T_S + 2e_1/N \qquad (2.14)$$

where H_t is the necessary operation time for editing program and N is a number of the data channels. $1/T_F$ and $1/T_S$ correspond to the numbers of occurrences of both editing and segment transfer requests in one second. The third term corresponds to the time stolen by the data channels to both the magnetic drum and the synthesizer part. In the computer system used (N=3, H_t=1.5 msec, W_t=400 μsec), CPU spent time for speech output is about 5 %.

(5) Access time to the magnetic drum

An editing request occurring at the time when one of $A_i(B_i)$ becomes empty, must be processed before the next request occurs (i.e., $B_i(A_i)$ becomes empty). Otherwise, the resulting output speech sounds are inevitably interrupted with unexpected pause and degraded in quality. The maximum number of multiplicity of the system is heavily dependent on the access time of the magnetic drum, because the mean duration of a dyad segment sequence in $A_i(B_i)$ is about 110 msec, and the access time is about 10 msec. This has turned out to be a bottleneck point among the system components.

2.7.4 Features of the System

Among various kinds of speech output systems, the following are notable advantages of the system.

(1) It can produce speech sounds of arbitrary vocabulary with high intelligibility.

(2) It can be made available for simultaneous usage of many users. The maximum number of multiplicity of the system (available number of simultaneous outputs) is about 10. It mainly depends on the mean access time of the magnetic drum. Installation of the faster magnetic drum or scheduling on it makes it possible for more available multiplicity to be achieved. CPU spent time increases at the rate of about 5 % as the number of multiplicity does.

In the system implemented by Nakata et al., it is reported that their system can be easily implemented for simultaneous usage of about 100 users. Controlling information for each output channel must be previously constructed in a file. If the pre-editing time of it is taken into account , the number of simultaneous usages will be remarkably limited.

In the system reported by Matsui et al., the maximum number of multiplicity of their system is nearly 20* when the parameter string for each input sentence is prepared in advance.

(3) Necessary memory size to store all of the compilation elements in the magnetic drum is about 170 kbits. It is smaller than Nakata's (800 kbits) and Matsui's

(1600 kbits)*.

(4) It can produce speech sounds in a very compact way by using zero-crossing wave. Digital technique is easily applied to construct the speech synthesizers. One of these is made of only 180 gates of integrated circuits.

(5) It can be easily connected to any type of computer system. A small version of it can be installed economically in a minicomputer system and will play an important role in growing applications of man-machine communication.

While, disadvantages of the system are as follows:

(1) Speech output somewhat lacks in naturalness. It is mainly due to the intrinsic characteristics of zero-crossing wave.

(2) Information amount to be controlled is of the rather larger value of 9 kbits/sec, compared with Nakata's (2.4 kbits/sec), Matsui's (0.4 kbits/sec)* and other models of speech synthesis method.

(3) Core memory capacity required for output double buffers is 512 x 2 chs . This is rather larger than others. But, it can be reduced to about 140 x 2 chs (L_F x 2) by using effective dynamic memory allocation.

* Estimated value

2.8 Conclusion

In this chapter, outline of the multiple speech output system and its system evaluation have been presented.

The experimental system could produce simultaneous speech outputs (10 or more) of arbitrary messages. The output speech sound was not so natural, but had enough intelligibility for human listening. A set of compilation elements adopted in the system consisted of dyad segment sequences of about 180 Japanese consecutive phonemes. But, if every V-C-V context is selected as a set of compilation elements, a maximum number of multiplicity will be increased. A necessary memory capacity for storing these data and output double buffers amount up to 700 kchs and 600 chs respectively.

Various statistical data occurring in the system have been shown on which the load effect of the computer and the maximum number of multiplicity were strongly dependent. CPU spent time on the speech synthesis was about 5 % for each active user.

From the statistical data, it was found out that one of the bottlenecks was the access time to the magnetic drum.

In order to evaluate the system performance precisely, system simulation should be conducted. And effective processing against the system's bottlenecks must be investigated by which the congestion in the system can be reduced as much as possible. These considerations by simulation will be dealt with in a detailed description of the next chapter.

III Queueing Models of the System and its Simulation for System Evaluation[36,37,38)]

3.1 Introduction

This chapter discusses the results of evaluation on the multiple speech output system and the way for improving performance of the system.

If many synthesizers are supposed to be connected to the system, various kinds of queues are formed. These include the queues for editing requests, segment transfer requests and OXI data requests in the synthesizer part. While, only one CPU installed in the computer executes many kinds of processings such as the editing program, the I/O control program, background programs and so on, in an order of pre-assigned priority.
A precise queueing model of this system is classified as a moving server queueing model with priority. It is, however, nearly impossible to analyze this highly complicated system in a mathematical way. Therefore, it is most realistic to deal with the simplest model containing the system's bottlenecks, and to show the way to break them for improving the system performance. In the multiple speech output system, the result of preliminary calculation in Section 2.7 shows that a bottleneck point is the queue for editing requests. Results of simulation on this point in real-time mode, are considered in those aspects such as the load effect of the computer,

maximum number of multiplicity and the required size of output user buffers.

In order to improve performance of the multiple speech output system, particularly on the maximum number of multiplicity, two kinds of queueing models, i.e., a cyclic queueing model and a moving server queueing model, are described and compared on their simulation results. A cyclic queueing model generalizes the structure of the system described in Chapter II. Load of the computer would be distributed in wide range, due to the random occurrence of editing requests to the computer. While, in a moving server queueing model, each active user is allocated a fixed amount of buffer. The computer supplies a dyad segment sequence with each buffer until it does not leave enough space to store further data. The maximum number of multiplicity would be increased at the cost of heavier loading of the computer.

Many kinds of techniques will be also available in these queueing models in order to increase the system performance up to the limit. The following techniques will be discussed, and reviewed on their effects from the simulation results. Those are; (1) a scheduling for accessing a secondary memory and multiple recording on the data file, (2) priority scheduling, and (3) rearrangement of data on the file suitable for faster information retrieval.

In all the cases, the system simulation is performed in on-line, real-time mode. Of many synthesizers of the system, two are actually constructed in hardware and others are simulated by a program and simple devices.

3.2 Queueing Models of the System

3.2.1 Performance Criteria

In the system evaluation, any selection of performance criteria is dependent on a special system to be evaluated. In the multiple speech output system, the following criteria should be picked up, because the system is operated in real-time mode to produce continuous speech sounds.

(1) Queueing error rate

This is the mean rate of unexpected pause of the output speech sounds in one second. It is induced by delay of real-time processings due to the overload of the computer and/or too long waiting time in a queue. Take for instance, as described in Section 2.7.3, an editing request which must be processed before the output double buffers become empty. If the editing request is not processed in a required time, the resulting output speech sounds will be inevitably interrupted with unexpected pause. With increase of the queueing error rate, quality of the output speech sounds is remarkably degraded. Then, the maximum number of multiplicity directly depends on the level of the queueing error rate.

(2) CPU spent time

In the system, many kinds of programs are running. The CPU spent time is a rate of the time in which the CPU is occupied by the speech production to the total effective time. The CPU is occupied by the editing program, the real-time monitor, and the I/O control program. The

less it is, the more the CPU is used by other processings.

(3) Capacity of output user buffers[39]

It is very important to estimate necessary buffer size, in such a hard real-time system as this one.

3.2.2 A Cyclic Queueing Model[18]

A cyclic queueing model is illustrated in Fig.3-1. It consists of a main queue and n feedback queues (user queues). The queue size of the user queue is limited to m . A request of the user (i) leaving the main queue will cyclically re-enter through the pre-assigned user queue (i) and wait until the preceding ones are fully processed. Input process to the main queue is the pooled output process of the user queues. The system in Chapter II is approximately expressed as one of these cyclic queueing models. The main queue corresponds to the queue for editing requests, and the distribution of its service time is almost similar to that of access time of the secondary memory which stores all of the dyad segment sequences. Limited size of the user queue m corresponds to a number of output user buffers. It is limited to a capacity of two when the output double buffers are used as described in Chapter II.
The distribution of service time of the user queue is similar to that of the actual duration of the dyad segment sequence (its mean value ; T_F=110 msec). An editing request is set in the main queue when one of the output user buffers becomes empty. Queueing error rate is

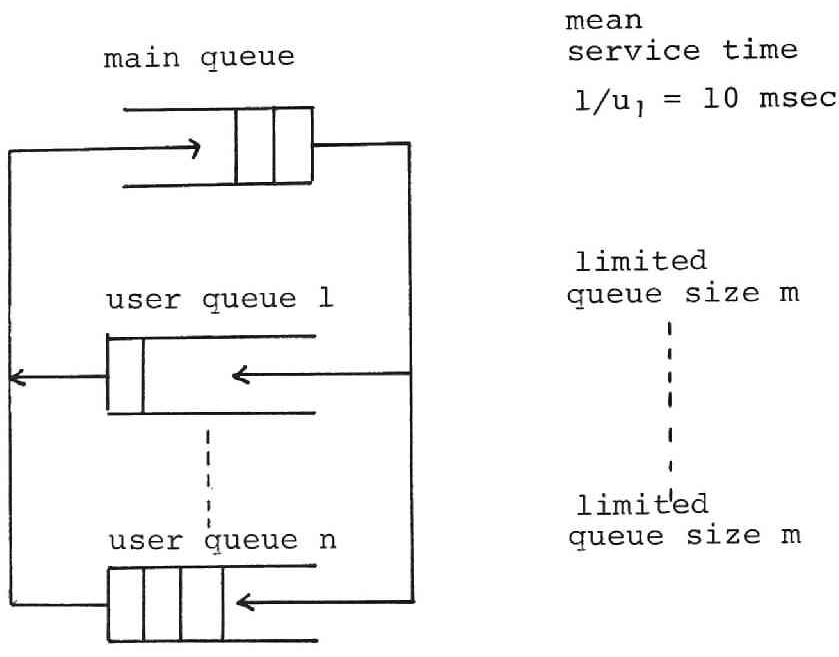


Fig. 3-1 Cyclic queueing model

expressed as the probability of the idle time of the user queue. The CPU spent time is the probability that the main queue is not idle. n corresponds to the number of multiplicity.

In this queueing model, random occurrence of independent requests from each user queue makes the main queue widely distributed in its length. That is, the computer to process the main queue is heavily loaded at one time, but at another time it remains idle.

In order to reduce idle time of the user queue, many techniques are available. These include a priority scheduling on the main queue, implementation of faster service time, and so on. In the model of Fig.3-1, each request carries its own user number and must enter into its own user queue. To this sort of model, queueing theory is not so powerful. An approximate model is shown in Fig.3-2, where n user queues are assembled into one feed-

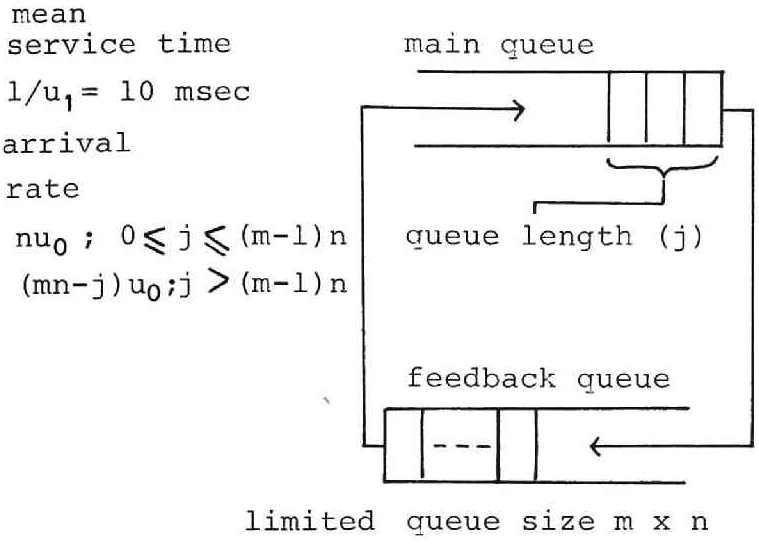


Fig. 3-2 Approximate model

back queue with limited size of $\underline{m}$ x $\underline{n}$.
Arrival rate to the main queue depends on the length (j) of the main queue (states); if $0 \leqslant j \leqslant (m-1)n$ then nu_0 , otherwise $(mn-j)u_0$. Here, u_0 is the mean service rate (per sec) of the user queue,and the number of requests waiting in the main queue and being served is defined as queue length. Change of arrival rate is due to the fact that if $j > (m-1)n$, then the requests probably occur only from (mn-j) non-empty user queues in Fig.3-1.

P(j) means the probability that the queue length in the main queue is just (j). $1/u_1$ is the mean service time of the main queue. The following equilibrium equations must hold (see, APPENDIX).

$$
\begin{aligned}
&u_1 P(1) = n u_0 P(0) && ; \quad j=0 \\
&u_1 (P(j+1) - P(j)) = n u_0 (P(j) - P(j-1)) && ; \quad 1 \leqslant j \leqslant (m-1)n \\
&u_1 P(j+1) - ((mn-j)u_0 + u_1) P(j) + (mn-j+1) u_0 P(j-1) = 0 && ; \quad (m-1)n < j < mn \\
&u_1 P(mn) = u_0 P(mn-1) && ; \quad j=mn
\end{aligned}
\tag{3.1}
$$

These equations can be solved in an iterative manner as follows:

$$
\begin{aligned}
&P(0) = 1/A_m && ; \quad j=0 \\
&P(j) = (np)^j / A_m && ; \quad 1 \leqslant j \leqslant (m-1)n \\
&P(j) = n! n^{(m-1)n} p^j / (A_m (mn-j)!) && ; \quad (m-1)n < j \leqslant mn \\
&A_m = 1 + \sum_{j=1}^{(m-1)n} (np)^j + n! (np)^{(m-1)n} \sum_{j=1}^{n} p^j / (n-j)!
\end{aligned}
\tag{3.2}
$$

When the length of the main queue is greater than $(m-1)n+1$, the particular request of the user (i) does not exist in the feedback queue. Queueing error rate is calculated from the probability that no requests of the user (i) exist in the feedback queue.

$$
\begin{aligned}
\text{Queueing error rate} &= \sum_{j=1}^{n} P((m-1)n+j)\, {}_{n-1}C_{j-1} / {}_{n}C_{j} \\
&= (n-1)! (np)^{(m-1)n} \sum_{j=1}^{n} j p^j / A_m (n-j)!
\end{aligned}
\tag{3.3}
$$

$$
\text{CPU spent time} = 1 - 1/A_m \tag{3.4}
$$

Here, p equals u_0/u_1.

Fig.3-3 shows the results of calculation, in case of p=0.06,0.08 and 0.12. As Fig.3-3 shows clearly, the increase of the limited size of the user queue m, makes a reduction of the queueing error rate to some extent. But, its effect gradually decreases as m rises. Increase of CPU spent time is independent of the values of m. Figs.3-4,5,6 and 7 show the probability distribution of the queue length and the mean queue length of the

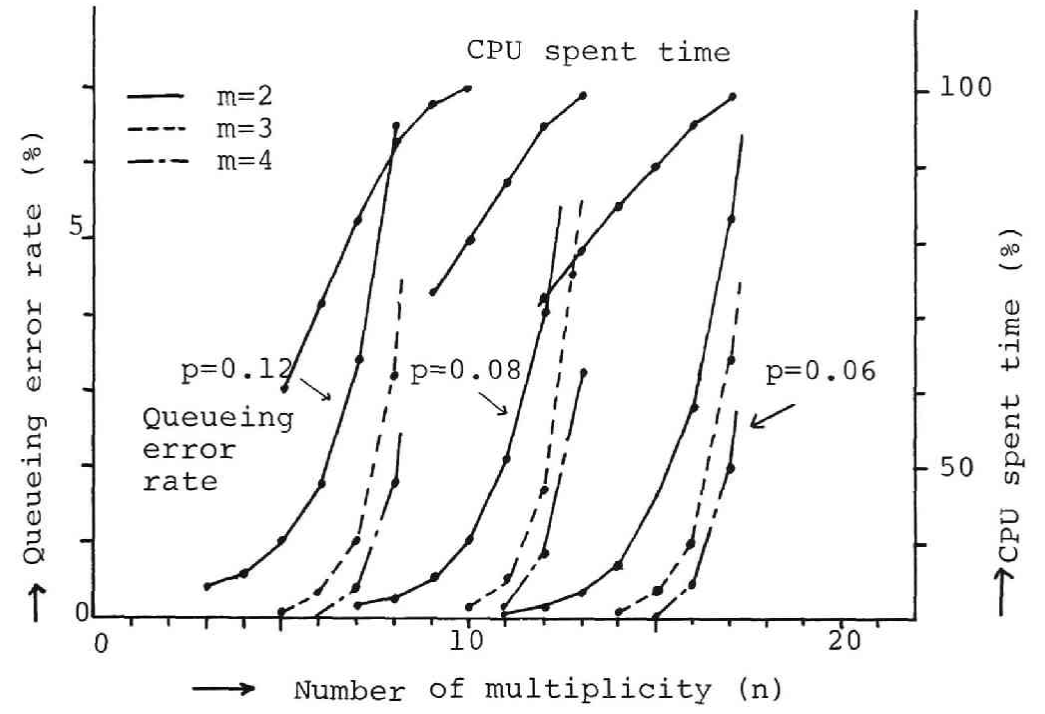


Fig. 3-3 Results of calculation for the cyclic queueing model

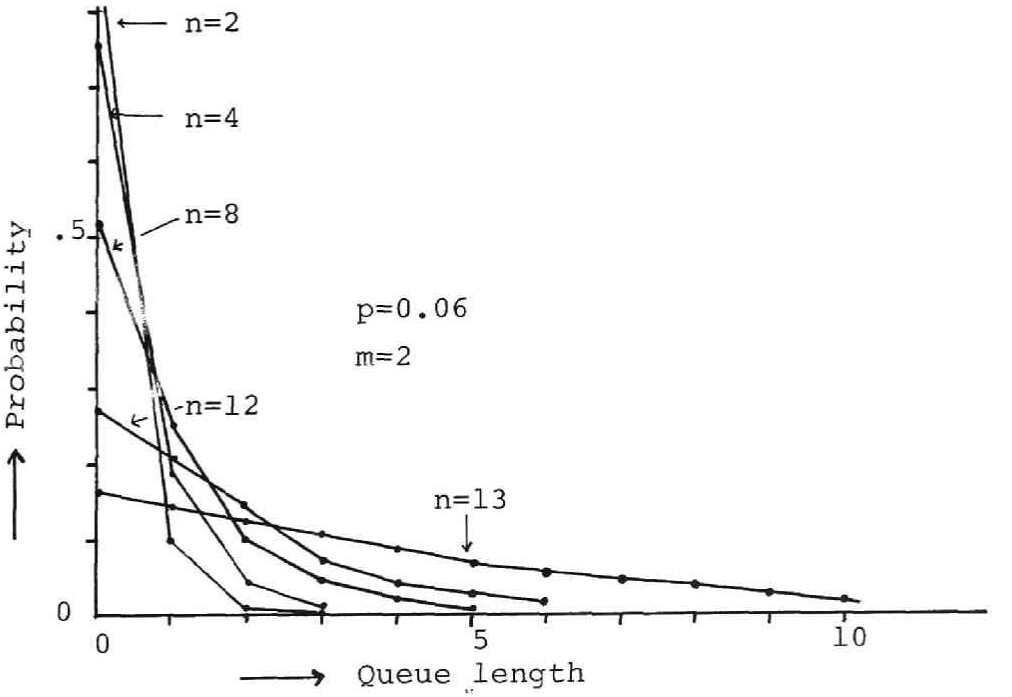


Fig. 3-4 Probability distribution of the queue length in the main queue

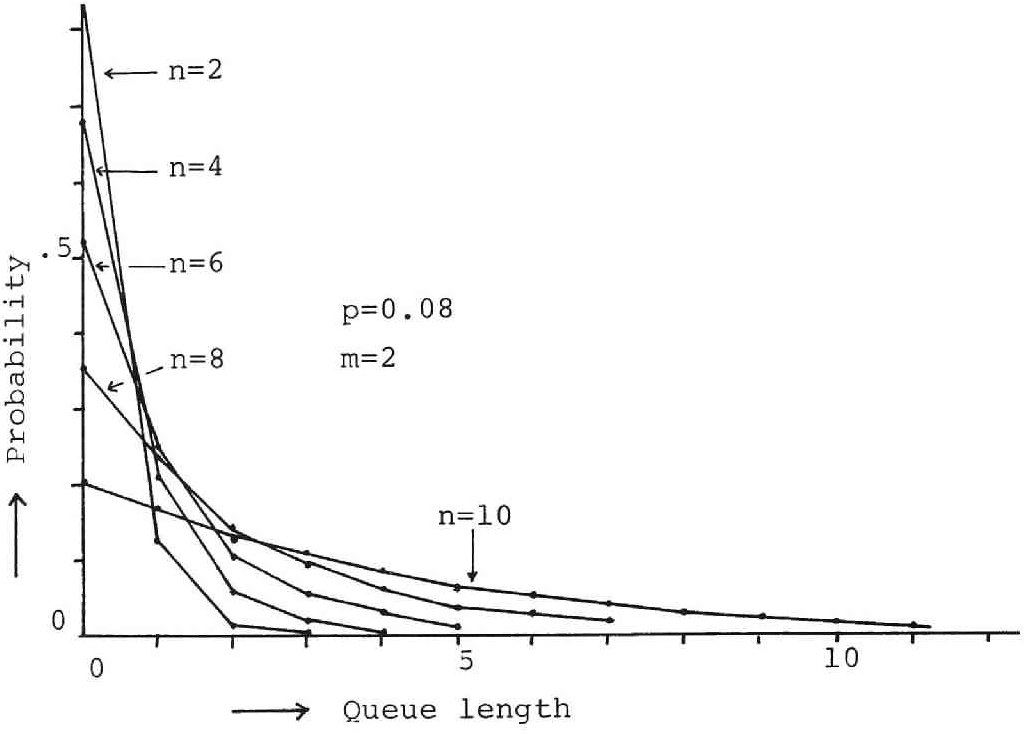


Fig. 3-5 Probability distribution of the queue length in the main queue

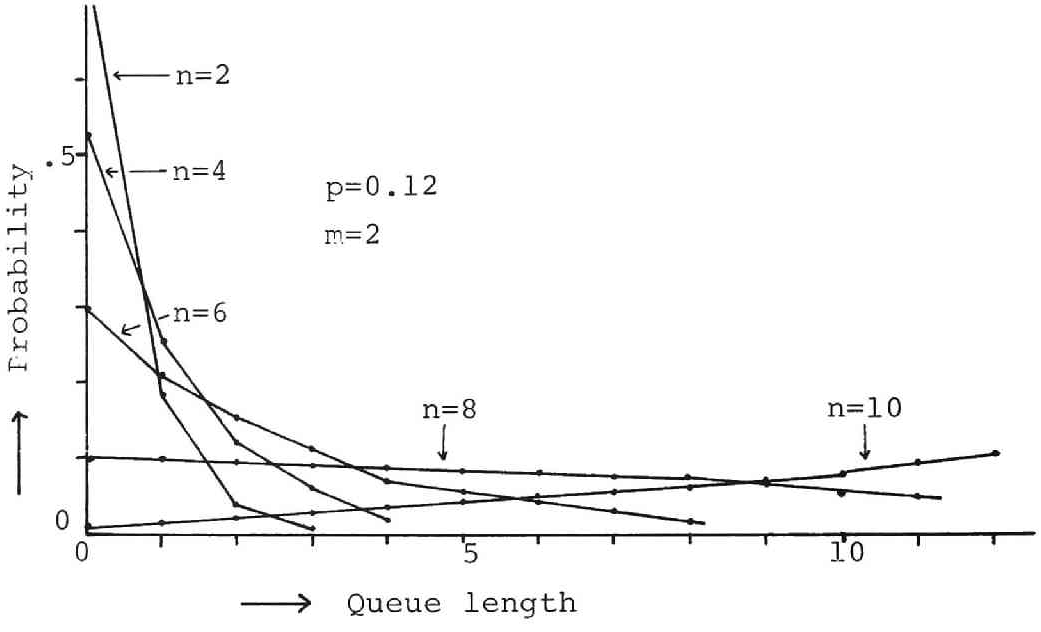


Fig. 3-6 Probability distribution of the queue length in the main queue

main queue as a function of n, respectively, where m is set to be equal to two. With the increase of n, both the mean queue length and the queueing error rate sharply rise. Queue length is widely distributed. Even when the queueing error rate shows rapid increase, the idle time probability of the main queue is still not so small.

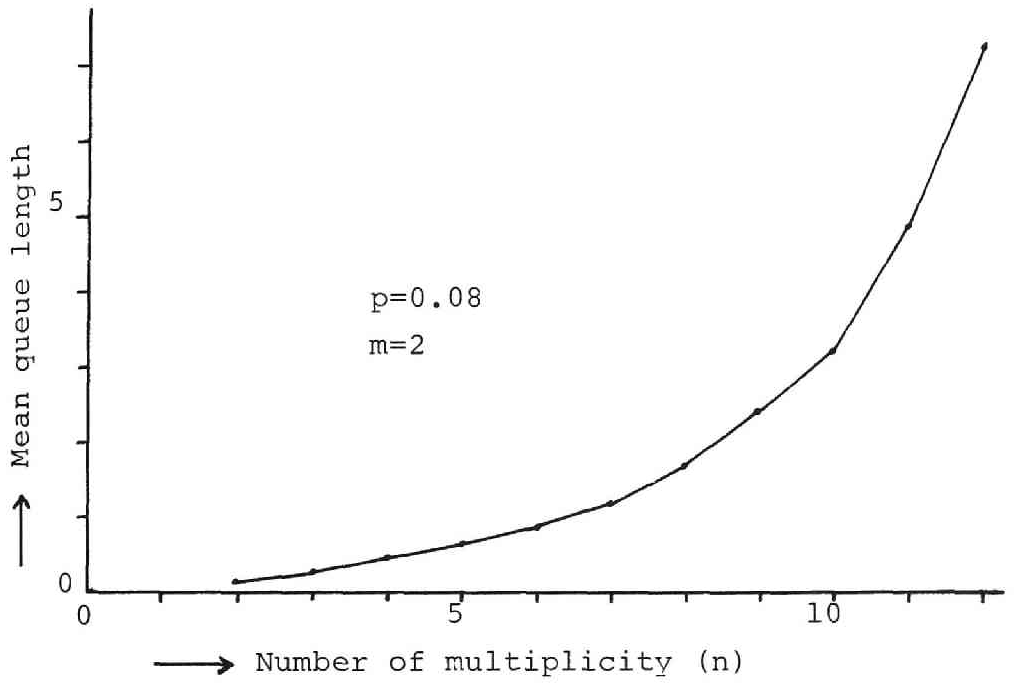


Fig. 3-7 Mean queue length in the main queue

3.2.3 A Moving Server Queueing Model[40,41,42]

In the cyclic queueing model, the queue length is widely distributed in range. For instance, at n=10 in Fig.3-5, the probability of the idle time of the main queue is nearly 0.2. But, relatively long queues also occur to result in an increase of the queueing error rate (about 1 % in Fig.3-3). In order to reduce it and then increase the maximum number of multiplicity n, the idle time of the main queue must be managed to be effectively used as much as possible. But, the requests in the cyclic queueing model are set into the queue just at the time of their occurrence. Therefore, some of the requests must be set in the queue prior to their actual occurrence.

Instead of doing this, in a moving server queueing model, a computer (server) can transfer a dyad segment sequence to each user buffer, which is fixed in its size,

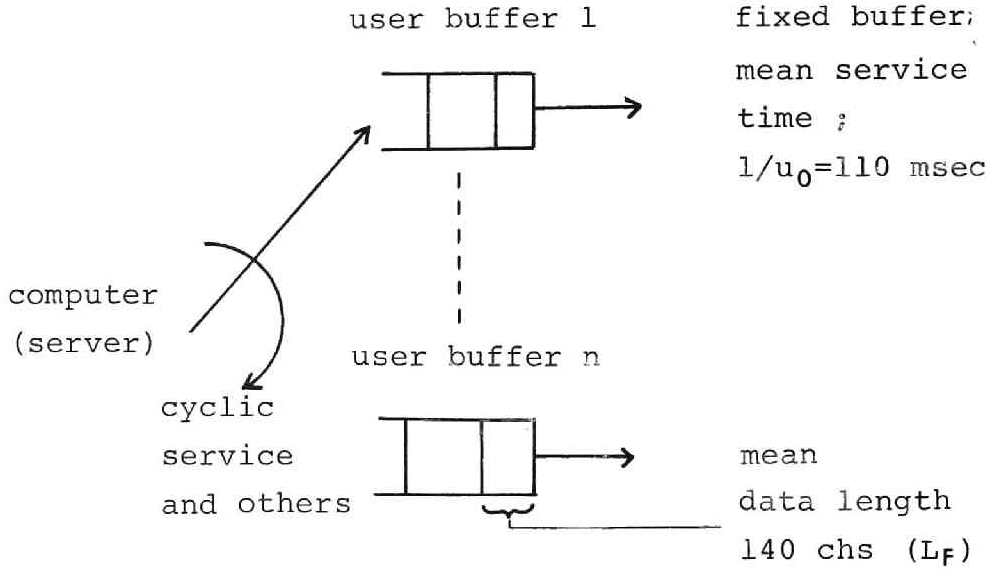


Fig. 3-8 Moving server queueing model

cyclically if the allocated user buffer has enough space to store data, as shown in Fig.3-8. When buffers of all users become full, the computer operates on the background programs and waits till any one of the user buffers can be available for data transmission. The order of processing for each user buffer is usually in a cyclic way. Priority processings are, however, also available in such a case of warning condition that some user buffers are about to be empty. It is also powerful to schedule n requests in a suitable order for the processing of the computer (e.g., accessing the secondary memory).

Mean service time of the user buffer corresponds to the actual duration of the dyad segment sequence T_F (about 110 msec). Distribution of data length entering into the user buffer is similar to that of dyad segment sequence as shown in Fig.2-14.

3.3 Characteristics of the Secondary Memory[43)]

In the I/O limited system, some techniques may be utilized in order to decrease the physical access time to the secondary memories. It is very useful to know the characteristics of them, for system designing. In this section, a magnetic drum and disk, are dealt with for preliminary estimation of the system performance in the following sections.

(1) Magnetic drum

(a) configuration

The magnetic drum used, is divided into 64 bands of fields, one of which includes 4 tracks in parallel. The angular coordinate of each track is divided into 10 sectors.It can be accessed by specifying the addresses of both a band and a sector. Its rotation time T is about 18.6 msec.

(b) calculation of the access time

Suppose that k requests are ready to access the magnetic drum. According to the first-in-first-out discipline (FIFO, requests are scheduled in an order of their arrivals), the total access time required W is,

$$W=kT((R+2)/2R) \quad (sec) \tag{3.5}$$

where R means the number of records on each track.

On the other hand, if k requests are scheduled according to the shortest-access-time-first (SATF, a request whose location on the moving surface is nearest to the present read-head, is selected), then W is expressed approximately for large k,

$$W=T(1/2+(k+1)/R) \quad (sec). \tag{3.6}$$

These equations conclude that SATF scheduling will reduce the access time to a greater extent.

(c) the access time verified by the experiments

Fig.3-9 shows the total access time required of the magnetic drum. The editing program and the data file described in the previous chapter are in use. The total access time for k editing requests on FIFO is proportionate to k, and expressed by,

$$W=10\ k \quad (msec) . \tag{3.7}$$

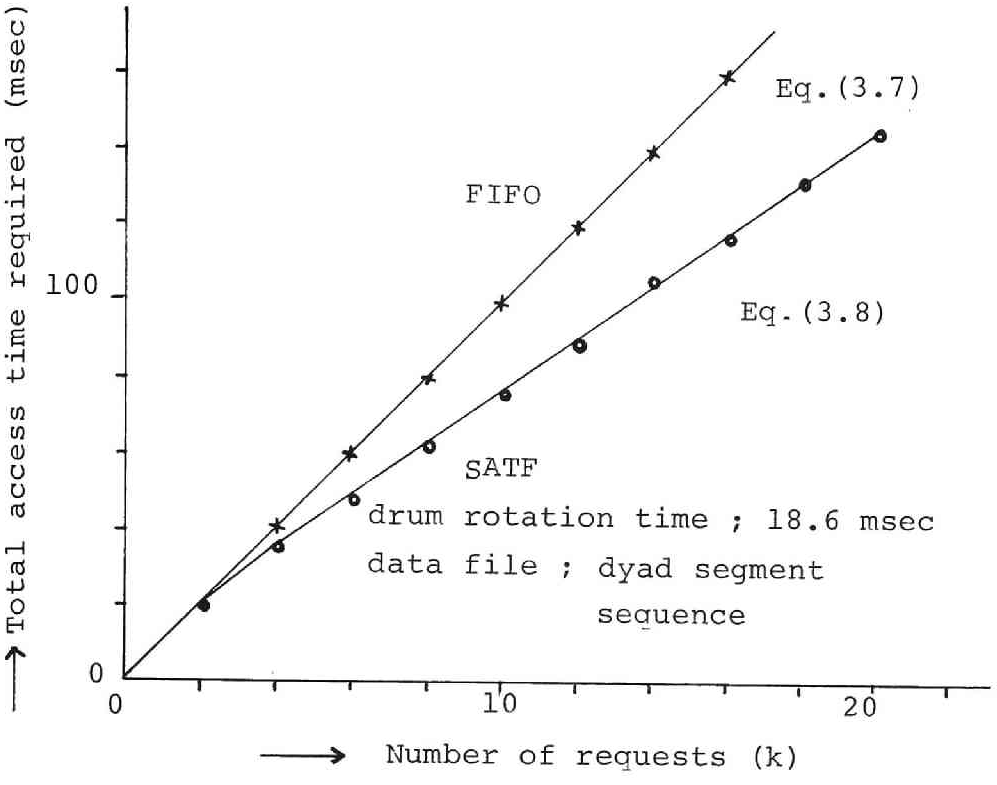


Fig. 3-9 Access time of the magnetic drum

On the other hand, if the access is based on SATF, $\underline{W}$ is,

$$W=(15.0+6.2\ k) \quad \text{(msec)}. \qquad (3.8)$$

(2) Magnetic disk

(a) configuration

The magnetic disk used is assembled by 10 disks of the same radii equally spaced along a common axis. The entire assembly is divided into 202 concentric sub-fields called cylinders. The cylinder is composed of 10 tracks. Each one of them is located at the intersection of the cylinder and each disk surface. The angular coordinate of each track is divided into 24 records. Then, the magnetic disk is

addressed by specifying the locations of a cylinder, a track, and a record. There is an assembly of movable arms equipped with read/write heads.The arms can not move independently of each other. Each time a new cylinder is requested, the arm assembly is necessarily to be re-positioned. The operation time of reposioning the arms is known as a seek time S. It depends on moving distance of the arm assembly, as shown in Fig.3-10.
It takes about 10 msec for the arm assembly to move to the adjacent cylinder. And the rotation time of the disk is about 25 msec.

(b) calculation of the access time

Suppose that k requests are ready to access the magnetic disk, two cylinders of which are only used. According to FIFO discipline, the mean value of the total access time is,

$$W=((R+2)T/2R+(P_{10}+P_{01})S)k \quad \text{(sec)} \qquad (3.9)$$

where $(P_{10}+P_{01})$ means the probability of transition from one cylinder to the other. The first and the second terms in right hand side of Eq.(3.9) correspond to the mean rotation time and the mean seek time of the magnetic disk, respectively. While, in case of SATF, W is approximately

$$W=T((R+2)/R+k/R) + S \quad \text{(sec)} \qquad (3.10)$$

for large k.

(c) the access time verified by the experiments

In Fig.3-11, the mean value of the total access time required for k editing requests is shown. In case of FIFO, W can be expressed by,

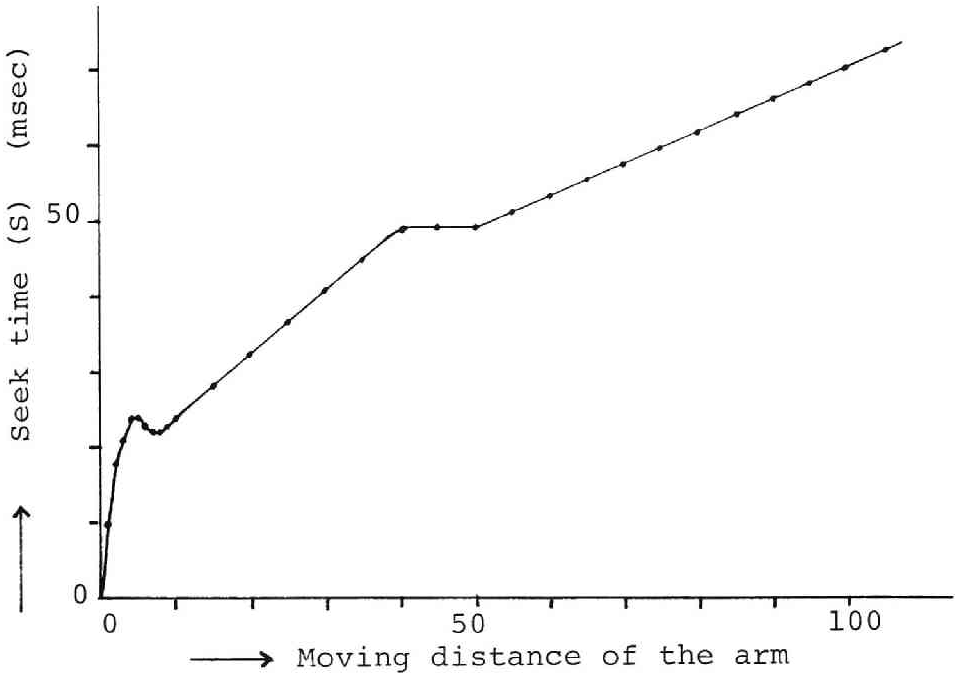


Fig. 3-10 Seek time of the magnetic disk

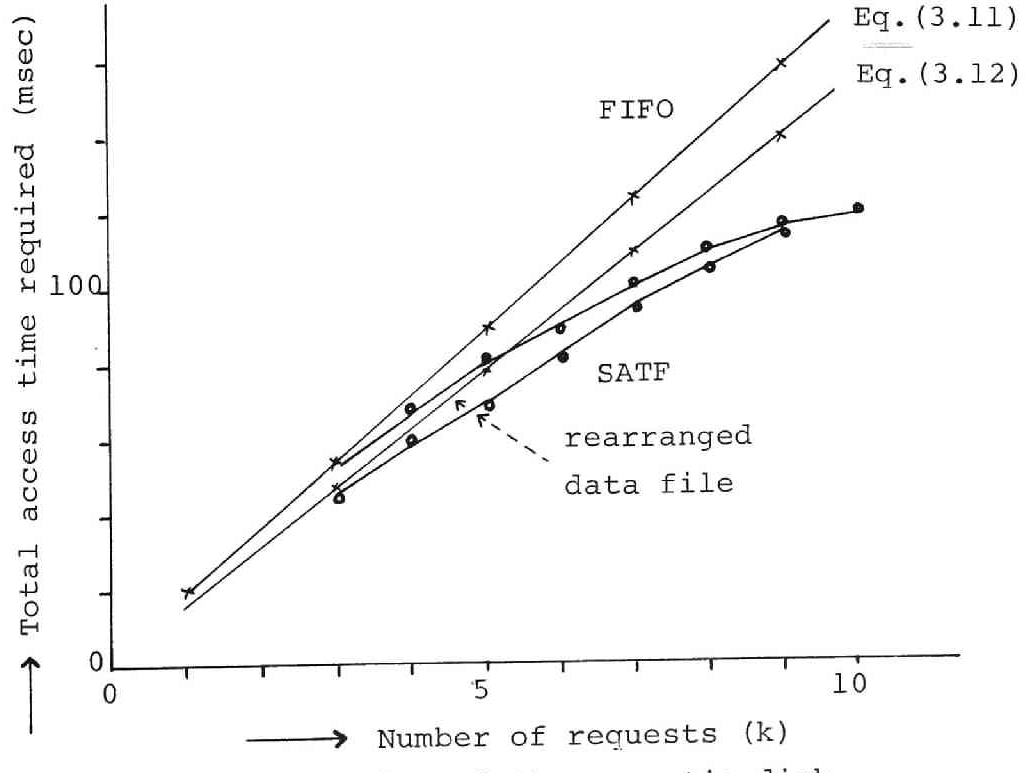


Fig. 3-11 Access time of the magnetic disk

$$W=17.6\ k \quad \text{(msec)} \tag{3.11}$$

where $(P_{10}+P_{01})$ is 0.2. If $(P_{10}+P_{01})$ is less, then the mean access time is proportionately reduced. It is done by rearranging data items on the file in an order of frequent use. Data items (i.e., dyad segment sequence) appearing more frequently are stored in the first cylinder, and others in the second. This rearranged data file reduces $(P_{10}+P_{01})$ to 0.05, and then the total access time results in,

$$W=15.6\ k \quad \text{(msec)}. \tag{3.12}$$

These results almost coincide with the previous calculation of Eq.(3.9). While, the total access time based on SATF is also shown in Fig.3-11. For large k, the access time rises at the rate of 3 or 4 msec as k increases by one. As k increases, the total access time required shows a great difference, as compared with the case of FIFO.

3.4 Simulation for the Cyclic Queueing Model

In order to raise the throughput of the system, many techniques will be used. One is based on the physical reduction of the access time of the secondary memory. This includes scheduling on it, multiple recording in quick bands, rearrangement of the data suitable for retrieval. The other is by operational methods. Typical examples are priority scheduling for urgent requests, and the increase in the number of user buffers m (i.e., user queue in

Fig.3-1). In this section, results of simulation in real-time mode, are reviewed on the effects induced by some kinds of improving techniques.

3.4.1 Simulation Method

The system simulation is performed in on-line, real-time mode. Of many synthesizers of the system, two are actually constructed in hardware and others are simulated by programs and simple devices. Many conditions can be supposed so that many active users would be connected to the computer, and a wide variety of timing conditions in real-time mode would be realized. Input sentences for each active user are punched in Roman letters and stored in each area of the core memory. They consist of hundreds of characters in size. In order to realize the simultaneous usage of n active users, these messages are repeatedly used to be produced into speech sounds for a certain amount of time.

Software for the simulation consists of a real-time monitor, an editing program, an I/O control program and a simulation program. The former three programs are quite similar as those of the previous chapter, so their description is omitted here.

Data of the duration of the successive dyad segment sequences, which are already completed and stored in each user buffer, are stored in the table $S1_i(S2_i,\ldots,Sm_i)$ allocated to each active user (i). Here, m means the number of the areas of the user buffers. These data are quantized by 10 msec. The content of $S1_i$ is reduced by one at every 10 msec clock of an interrupt signal.

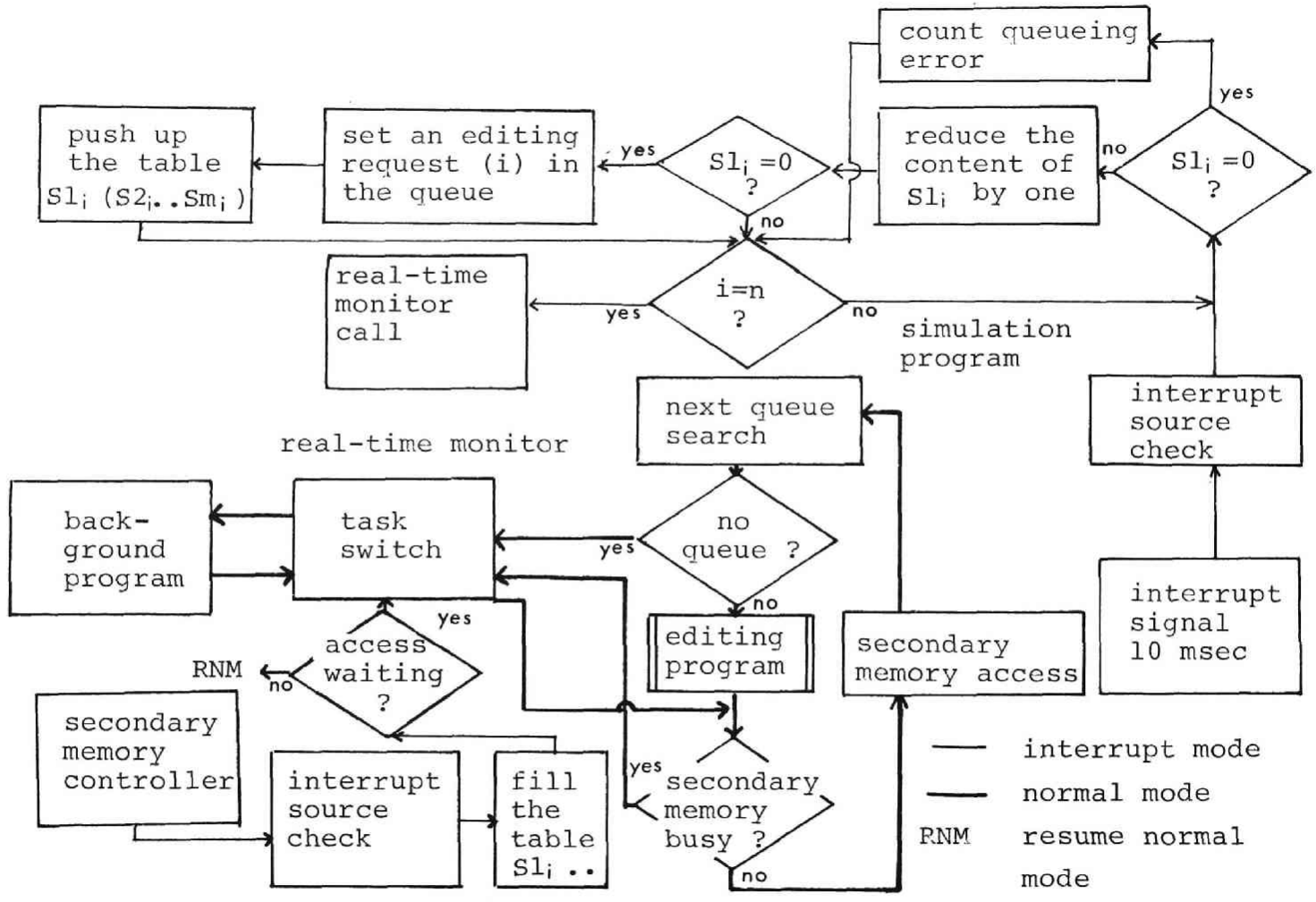


Fig. 3-12 Flowchart of the simulation for the cyclic queueing model

When Sl_i becomes empty, an editing request is set in the queue, and the table is pushed up. Behavior of the editing program is the same as in the system described in Chapter II except that it fills the table with the data of the duration of the dyad segment sequence which is just read out.
Queueing errors are counted at every interrupt signal, when neither of the user buffers has been prepared (i.e., Sl_i ($S2_i$,...,Sm_i) are all empty).

The background program periodically performs simple calculations whose operation time is preset by the program. CPU spent time can be computed by reducing from the total effective time of the computer, the total processing time during which the computer is running in the background program. Fig. 3-12 shows the flowchart of the simulation.

3.4.2 Results of the Simulation

(1) In case of the magnetic drum

Fig.3-13 shows the results of both the queueing error rate and CPU spent time. The buffering effect of m is also presented in the two cases of the output double buffers (m=2), and triple buffers (m=3). In case of m=2, queueing error rate is about 0.3 % at the maximum number of multiplicity (n=10), and increases rapidly with n. The errors are completely zero at n=10, where m=3.However, the effect of the increase of m is not so large as might be expected in the preliminary considerations in Section 3.2. In both cases, CPU spent time increases at the rate

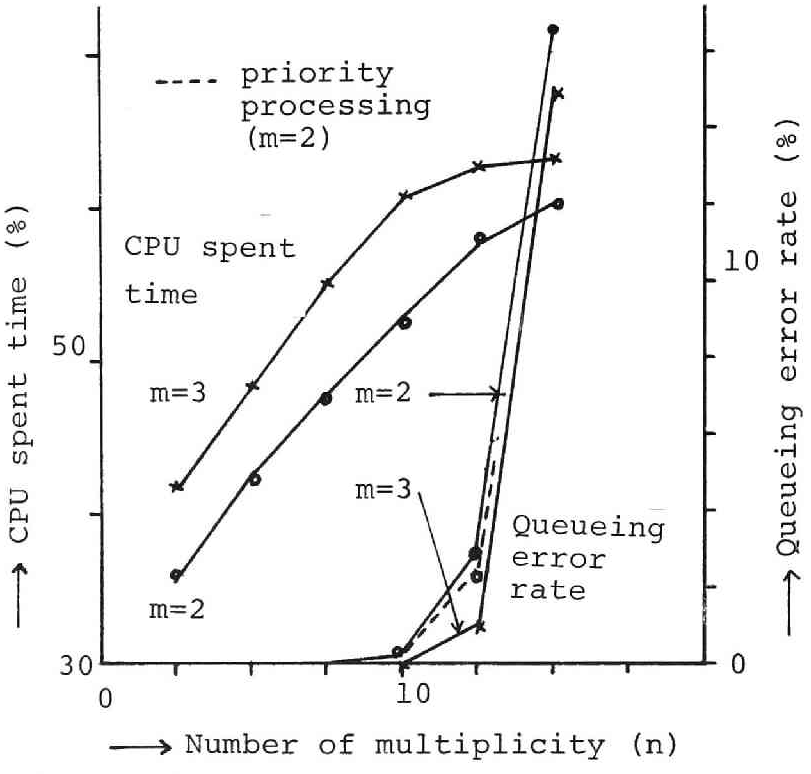


Fig. 3-13 Results of simulation (magnetic drum)

of 5.5 % as n increases, and saturates gradually. Fig.3-14 shows the distribution of the time interval when the queueing error continues.

The dotted line of Fig.3-13 shows the case of priority processing. The priority model is constructed of four level priority queues in which the waiting requests are rearranged to the higher level queue when the rest of time before the user buffers become empty, becomes less than a certain level. It is unfortunately quite similar to those in the previous case. But, the dotted lines of Fig.3-14 show that the variance of the interval when the queueing error continues is narrower.

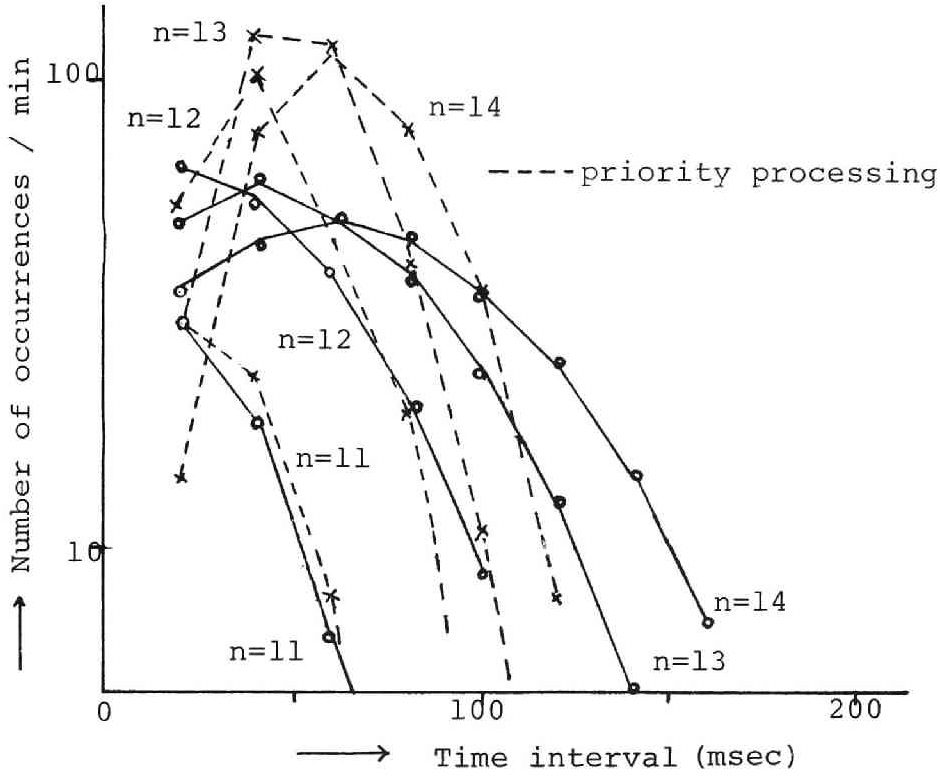


Fig. 3-14 Number of occurrences of time interval when the queueing error continues

(2) In case of the magnetic disk

Fig.3-15 shows the results of simulation for the case of double buffers (m=2). The queueing error rate rises rapidly at more than n=5. CPU spent time also increases at the rate of 5.5 %. Probability in which the first of the two cylinders of the disk is used is about 80 %. The CPU is only spent on the multiple speech output about 40-50 % of all the effective time, at the maximum number of multiplicity (n=4 or 5). This means that the overall processing time of the system is completely governed by the speed of the magnetic disk, and thus the system is operated in I/O limited form. Fig.3-16 shows the distribution of the queue length, which is sampled at

every 10 msec. In the region of low level queueing error rate (n=2 and 4) in Fig.3-16, probability distribution shows a rapid rate of decrease as the queue length increases. With increase of the queueing error rate, the mean queue length rises at a remarkable rate.

The dotted line of Fig.3-15 shows the results of the simulation, where the rearranged data file is in use. The number of multiplicity increases by one due to the actual reduction of the mean access time of the magnetic disk. In this case the access time is decreased by 2 msec for eack disk accessing as described in Section 3.3. Fig.3-17 shows the distribution of the queue length for the editing request.

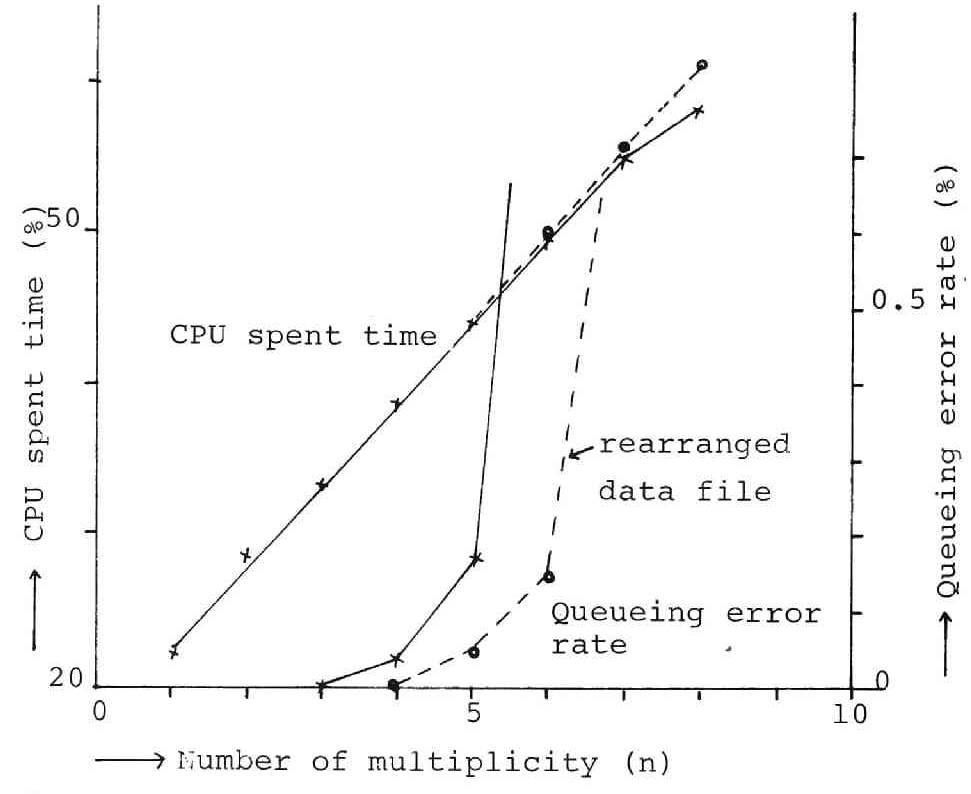
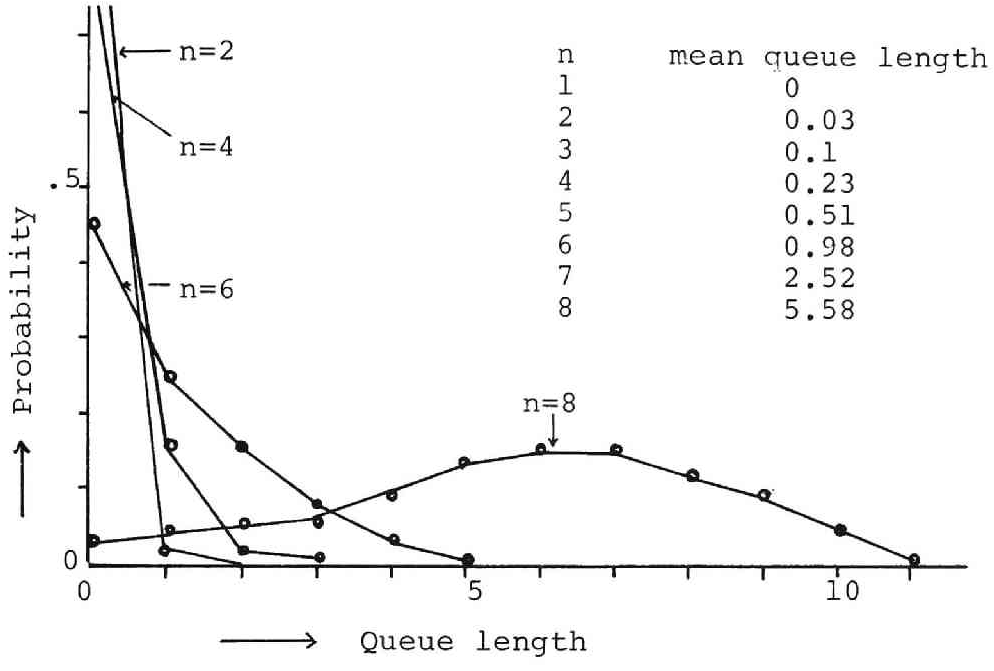
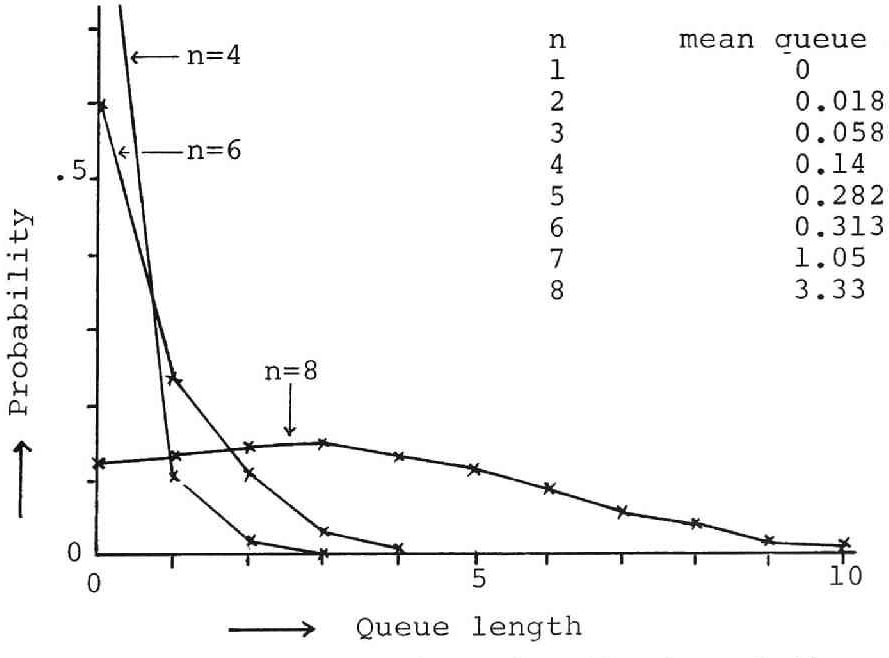


Fig. 3-15 Results of simulation (magnetic disk)



n	mean queue length
1	0
2	0.03
3	0.1
4	0.23
5	0.51
6	0.98
7	2.52
8	5.58

Fig. 3-16 Probability distribution of the queue length



n	mean queue length
1	0
2	0.018
3	0.058
4	0.14
5	0.282
6	0.313
7	1.05
8	3.33

Fig. 3-17 Probability distribution of the queue length (rearranged data file)

3.5 Simulation for the Moving Server Queueing Model

3.5.1 Simulation Method

(1) Cyclic service

The software for simulation consists of a real-time monitor, an I/O control program, an editing program and a simulation program. In this simulation, the magnetic disk is only used as a secondary memory. The simulation program is activated periodically by an external interrupt signal. Two tables are allocated to each active user. One (BS_i) indicates the amount of buffer size of the user (i) now in use. The other (BR_i) shows the time interval when the user buffer (i) does not become empty. The content of BR_i is quantized in 10 msec and reduced by one every 10 msec of the interrupt signal until it becomes zero. The content of BS_i is reduced by the amount of data length of a dyad segment sequence, at the time of its output. Queueing error rate is counted while the content of BR_i remains still empty.

The real-time monitor deals with the analysis of the interrupt signals, task switching among several programs, and the search of the active user ready for being processed in a pre-assigned priority. If the real-time monitor searches the active user whose BS_i is expected to have enough space for storing data, it makes the editing program run.

The obtained data of the length of the next dyad segment sequence will be accumulated in the BS_i unless the resulting content of BS_i exceeds the pre-assigned value (buffer size). Otherwise, the obtained data will be stored temporarily until the next chance of being

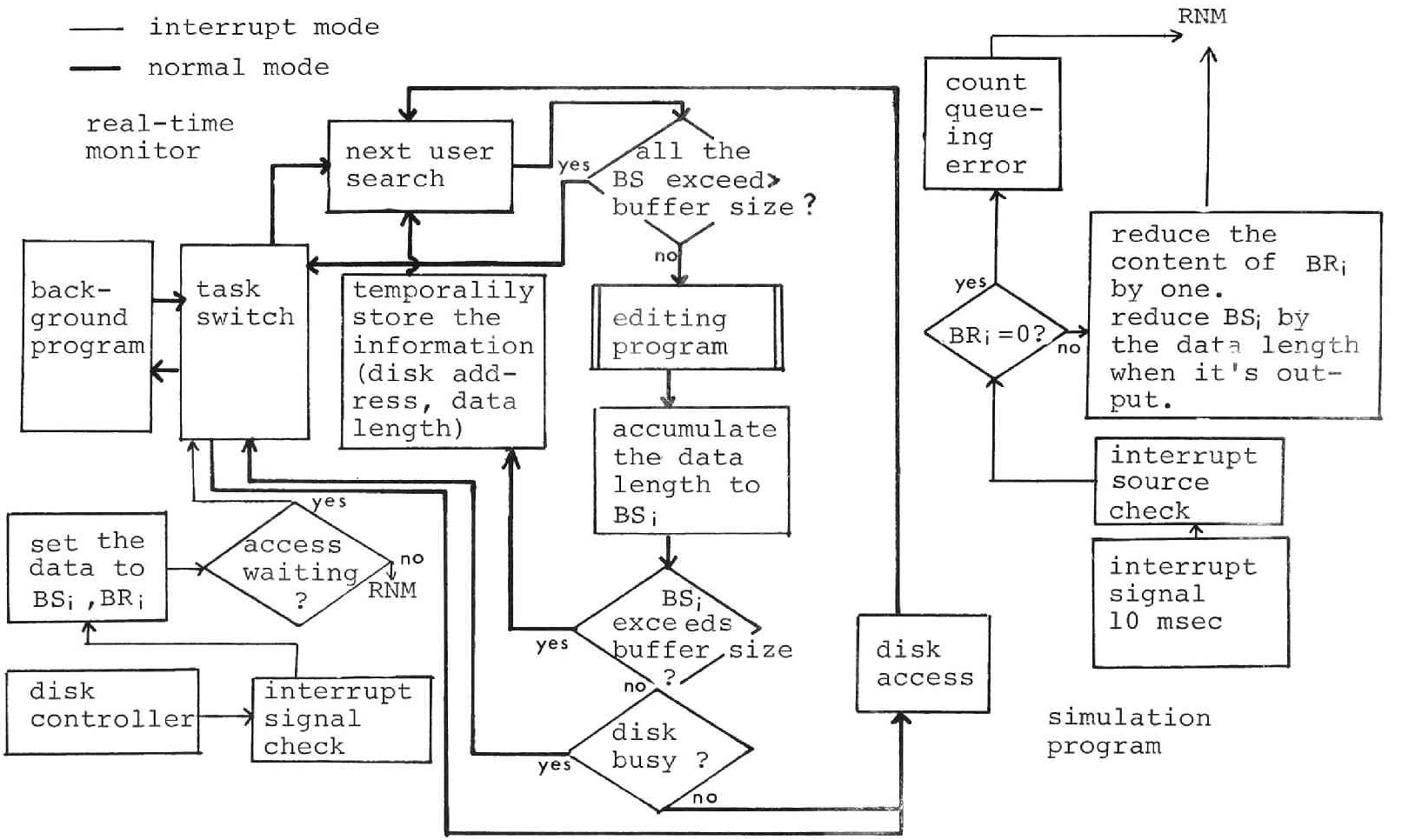


Fig. 3-18 Flowchart for the simulation of the moving server queueing model (cyclic service)

searched by the real-time monitor.

The editing program may contain disk accessing to get a dyad segment sequence. If the magnetic disk is busy, then the real-time monitor switches the task to the background program. The busy condition will be released at the end of the data transmission. And the interrupt signal from the magnetic disk controller will be accepted by the real-time monitor. The real-time monitor switches the task to the editing program for the disk accessing of the next request. The obtained data of the duration of the dyad segment sequence is accumulated in the BR_i. If all of the user buffers have no space for further data (see Fig.3-18, "all the BS exceed buffer size ?"), the real-time monitor switches its control to the background program. The flowchart of this scheme is shown in Fig.3-18.

(2) Priority service

Priority scheduling can also be realized. This queueing model is constructed of two level priority queues. The active user is allocated a high priority only when the number of dyad segment sequences in its user buffer BS_i becomes less than a certain level. The threshold is determined by experiments. The real-time monitor searches and processes the queues in an order of their priority.

(3) Scheduling on the magnetic disk

The magnetic disk used does not implement the readable counter by the software indicating the present

location of the read-head on the moving disk. This causes the scheduling somewhat difficult. The information on the address location supplied only by the end of the data transmission is not enough for the effective scheduling operations.

When a request directly accesses the magnetic disk, the following requests will wait until the busy period of the disk is released. During the waiting time, the real-time monitor switches its control to the editing program, instead of the background program as in the case of (1) , to process for another active user as much as possible. In this process, a queue is formed for each location which is 1/8 of the circumference of the disk (see Fig.3-19, "set disk access queues"). Requests associated with the same location field are set in the corresponding queue.

When the disk busy period is released, the real-time monitor can learn the location of the read-head, referring to the address register of the disk controller, and select that request from the corresponding queue which requires the shortest access time (SATF discipline). This selection must be done before the required location on the moving disk does not yet pass away under the read-head. But, the time required for this selection is governed by a probabilistic event, being dependent on the system conditions. This concludes that more precise division than 8 locations of the circumference of the disk is meaningless.

After the processing of the queues for one cylinder is completely finished, the arm assembly must be moved to the other to process the remaining queues.

In this scheduling of the SATF, attention must be paid since processing of an urgent request may lag behind.

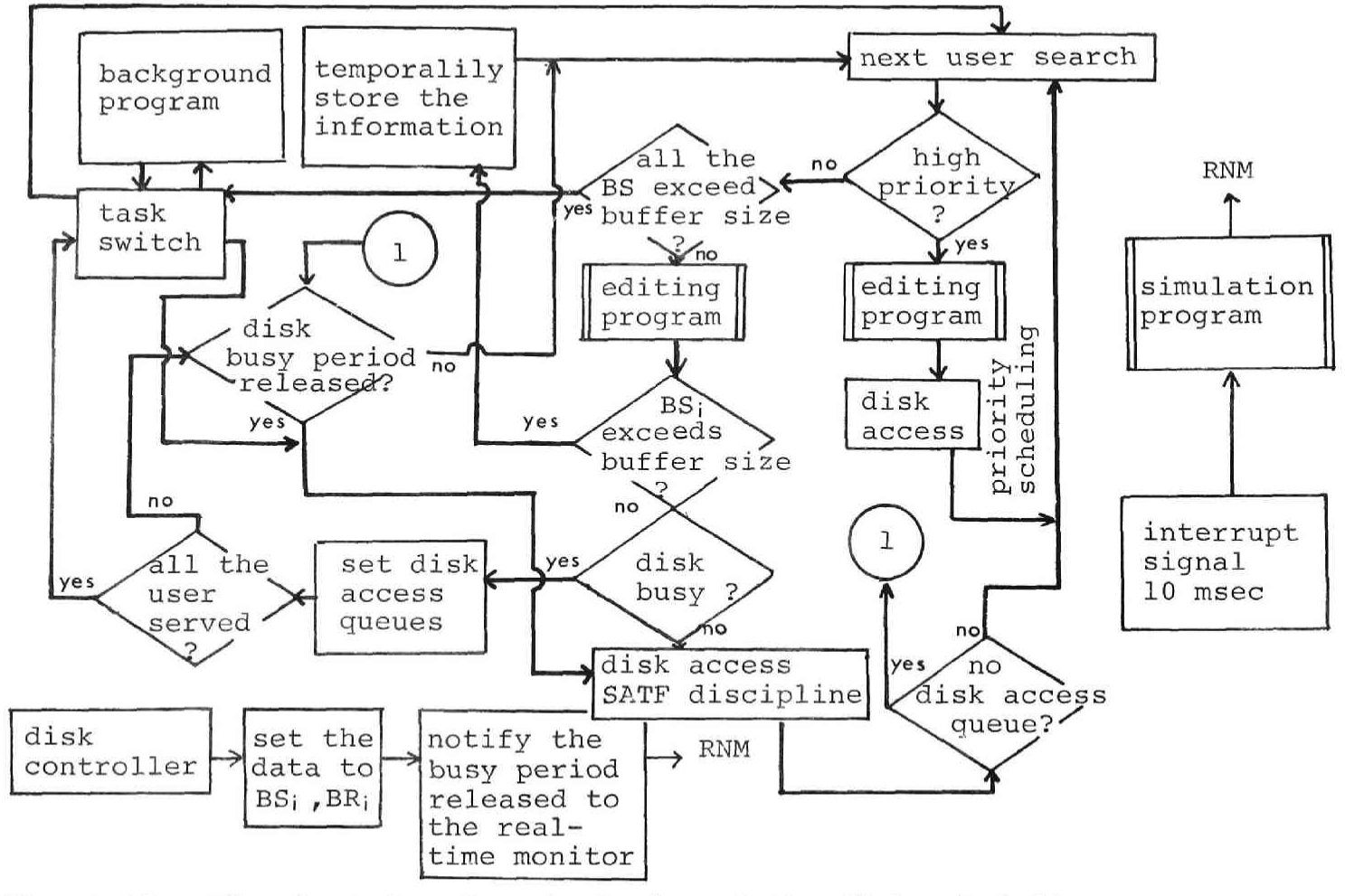


Fig. 3-19 Flowchart for the simulation of the disk scheduling

To prevent this situation, a priority scheduling is added to this scheduling algorithm. Here, the urgent request is not subjected to the disk scheduling , but can independently access the disk in an immediate way. Flow-chart of this scheme is shown in Fig.3-19.

(4) Quick bands

Multiple recording of data on the same track is one of the useful techniques for the realization of the faster access time. They are often referred to as quick bands. During the operation of the editing program, one of the multiple records is selected which is located in the shortest distance from the present position of the read-head. Necessary memory space for storing all of the dyad segment sequences is about one and a half cylinders of the magnetic disk. Multiple recording is limited to the data items of relatively large occurrence probability in order to limit the total amount of data within two cylinders.

3.5.2 Results of the Simulation

Fig.3-20 shows the results of the simulation in case of the cyclic service, where the size of the user buffer is limited to a capacity of 1 kchs and 2 kchs. In the case of 1 kchs, queueing error rate is almost zero up to n=6. But, for more than n=7, it increases at a remarkable rate.

In the case of 2 kchs, the queueing error rate is completely zero up to n=7. For more than n=8, it shows a rapid rate of increase. CPU spent time is almost equal in both cases. Effects of the user buffer size on the system performance are just as might be expected.

The results for priority scheduling are shown in the dotted line in Fig.3-20 where the threshold value of the number of dyad segment sequences in its user buffer is one. In the case of 1 kchs used, the queueing error rate is completely zero up to n=6. But, in the case of 2 kchs the priority scheduling has no effect. CPU spent time is nearly equal to that in the case of the non-priority queue.

As compared with the results of the cyclic queueing model described in Section 3.4, the moving server queueing model produces better effects on the maximum

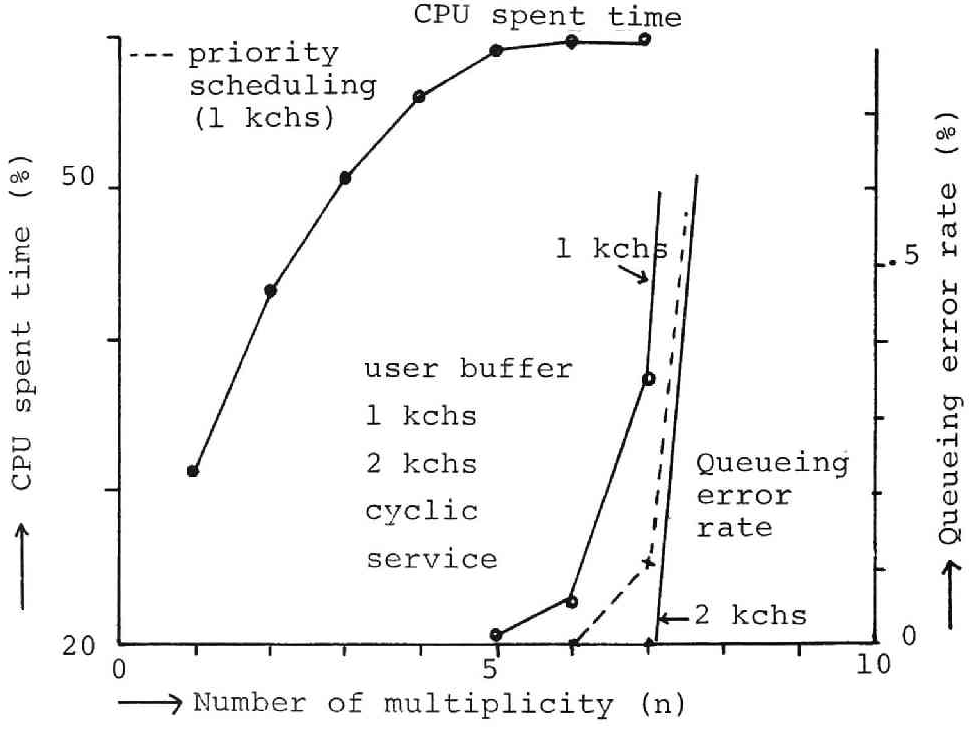


Fig. 3-20 Results of simulation (magnetic disk)

number of multiplicity. That is mainly due to the difference of the mathematical models, and the simple structure of the real-time monitor, which may not often analyze various timing conditions .
But, CPU spent time remains 60 % even at the maximum number of multiplicity. The whole system is in I/O limited form.

Results of scheduling on the magnetic disk are shown in Fig.3-21. Queueing error rate is about zero up to n=6 in case of 1 kchs used, and n=8 in case of 2 kchs. CPU spent time is nearly 80 % and is not dependent on the increase of user buffer size.

Fig.3-21 also shows in the dotted lines, the results of the quick bands. The maximum number of multiplicity is

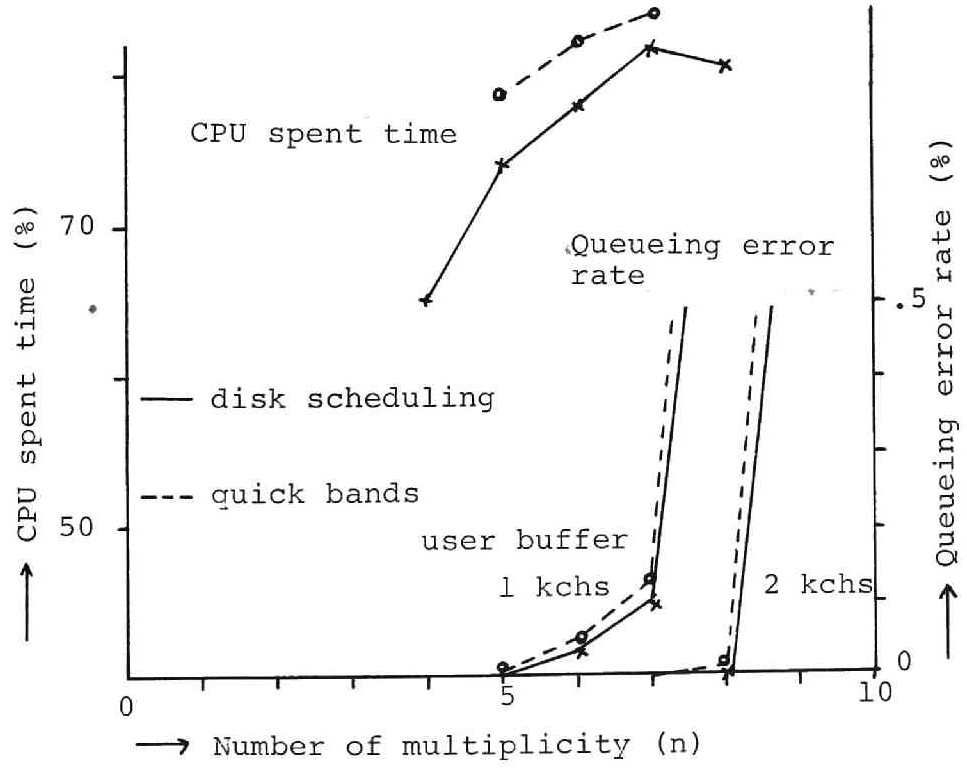


Fig. 3-21 Results of simulation (magnetic disk)

almost similar to that in the case of the scheduling on the magnetic disk.

Fig.3-22 shows the results of the simulation when the rearranged data file described in Section 3.3 is in use. In each case, the maximum number of multiplicity can be increased by one as compared with the previous cases when the user buffer is 1 kchs. But, in the case of 2 kchs, the rearrangement of the data file does not produce any effect, as shown in Fig.3-11.

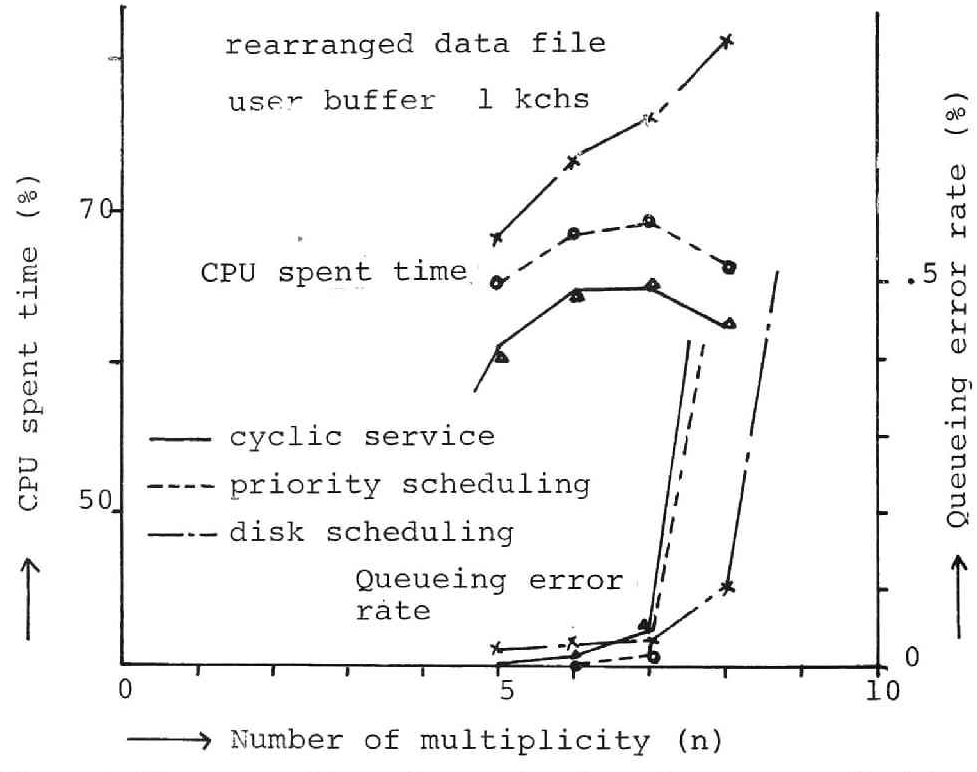


Fig. 3-22 Results of simulation (magnetic disk)

3.6 Conclusion

In this chapter were discussed not only the results of the system evaluation, but also some processing methods in which the maximum number of multiplicity was improved to some extent. Simulation by a program and simple devices created environmental conditions under which many active users were connected to the computer. A technique of software monitoring was used to gather some statistical data. An overhead time inevitably induced during the simulation was not so great that simulated behaviors of the system would be expected to give much information on the actual behaviors of the system in multiple usage. But, in order to simulate the system in more detail, a simulation by a computer complex must be used. If a small computer is operated for precise simulation of the synthesizer part, and connected to the main computer running on the multiple speech output programs, the load effect of the main computer is just the same as it would be in actual running.

The results of simulation showed that the simple replacement of the processing method improved system performance. For example, the number of multiplicity of the moving server queueing model associated with the disk scheduling was greater by 2 as compared with that of the cyclic queueing model.

IV Cyclic Queueing Process and its Control

4.1 Introduction

This chapter discusses a special control problem of a cyclic queueing model, which is composed of a main queue and n feedback queues. The reason why this model is considered here for controlling the queueing system is twofold. First, intrinsic characteristics of the model are such that the input process of the main queue is quite dependent on its output process, being connected through the feedback queues. This means that the control of requests before their arrivals to the main queue is easily achieved by controlling feedback queues. Second, this queueing model is available for controlling the multiple speech output system. The system configuration is different from that discussed in Chapter II. The system described here stores pre-edited dyad segment sequences for input sentences of a user in a numerical form on a cylinder of the magnetic disk. When one of the output double buffers becomes empty ,the computer is requested to send some units of dyad segment sequences. When other requests are in line, it happens that the data transfer lags behind and the queueing errors appear. In order to prevent this situation, it is a problem to determine dynamically how many units of dyad segment sequences should be read out at one time of disk accessing. A key concept of control is to reduce the queueing error rate as

much as possible with the smaller increase of the total memory capacity of output double buffers.

Results of both calculation and simulation indicate the effectiveness of a proposed control scheme.

4.2 Control of Cyclic Queueing Process

Suppose the cyclic queueing model shown in Fig.4-1. This model is constructed of a main queue, a controller, and n feedback queues whose size is limited to a capacity of two. Notable aspects are:

(1) A request of the user (i) leaving the main queue will again re-enter it through each feedback queue (i) pre-assigned to each user (i).

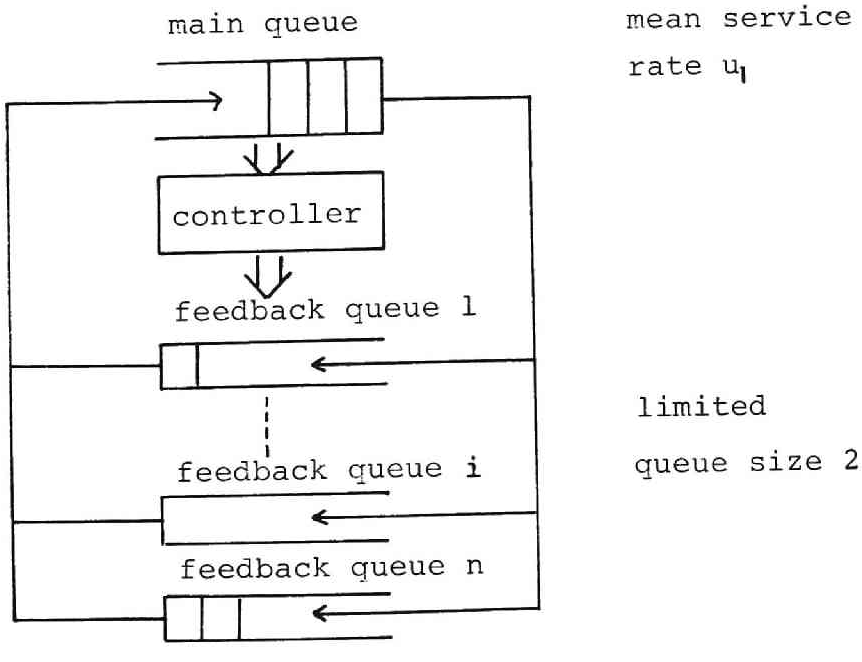
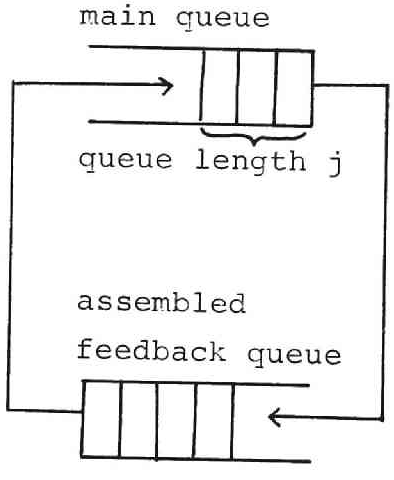


Fig. 4-1 Controlled cyclic queueing model

(2) Feedback queues act independently of each other and the pooled output of feedback queues forms the input process to the main queue.

(3) If the main queue is too long, a request set from a feedback queue must wait in the main queue for a long time until all the preceding ones are served. But, in certain cases, it would be more preferrable for a request to wait in a feedback queue rather than in the main queue. In that case, all the controller has to do is to degrade the service rate of the feedback queues.

Before considering the controlled queueing model, preliminary calculation for the uncontrolled cyclic queueing model will be given. The analytical tool of queueing theory is not so powerful for the case where requests are distinguished from each other, when entering the pre-assigned feedback queues. Then, some approximations are needed to solve it successively. An approximate model is shown in Fig.4-2, where n feedback queues are gathered



mean service rate
u_1

limited queue size $2n$
mean service rate
(arrival rate to the main queue)

nu_0/M ; $0 \leqslant j \leqslant n$

$(2n-j)u_0/M$; $n < j \leqslant 2n$

Fig. 4-2 Approximate model of the cyclic queueing model

into one assembled feedback queue with limited size of $2n$. Arrival rate to the main queue depends on the length (j)

of the main queue; if $0 \leqslant j \leqslant n$ then nu_0/M, otherwise $(2n-j)u_0/M$. Here, u_0/M is the mean service rate of the feedback queue, and $\underline{M}$ is a positive integer value. Change of the value of the arrival rate is due to the fact that if $j > n$, then requests supposedly occur only from $(2n-j)$ non-empty feedback queues. If both of the input and output processes are assumed as Poisson processes, the following equilibrium equations must hold.
Here, $P(j)$ means the probability that the length in the main queue is just (j), and u_1 is the mean service rate of the main queue.

$$
\begin{aligned}
& u_1 P(1) = nu_0/M \; P(0) && ; \; j=0 \\
& u_1 (P(j+1) - P(j)) = nu_0/M \; (P(j) - P(j-1)) && ; \; 1 \leqslant j \leqslant n \\
& u_1 P(j+1) - ((2n-j)\, u_0/M + u_1) P(j) + && \qquad (4.1) \\
& (2n-j+1) u_0/M \; P(j-1) = 0 && ; \; n < j < 2n \\
& u_1 P(2n) = u_0/M \; P(2n-1) && ; \; j=2n
\end{aligned}
$$

These equations can be solvable in an iterative manner as described in Section 3.2.2 and APPENDIX.

$$
\begin{aligned}
& P(0) = 1/A && ; \; j=0 \\
& P(j) = (np)^j / A && ; \; 0 < j \leqslant n \\
& P(j) = n! n^n p^j / (A \; (2n-j)!) && ; \; n < j \leqslant 2n \\
& A = 1 + \sum_{j=1}^{n} (np)^j + n! (np)^n \sum_{j=1}^{n} p^j / (n-j)! \; ; && \qquad (4.2)
\end{aligned}
$$

Here, $p = u_0 / (u_1 M)$. From the solution of Eq.(4.2), the following quantities can be obtained. Availability of the main queue is equal to one minus the value of the probability of the idle time $P(0)$. Then,

$$\text{Availability of the main queue} = 1 - 1/A \qquad (4.3)$$

Assembled feedback queue length is $(2n-j)$ which

occurs with a probability P(j). Then the mean value is,

$$\sum_{j=0}^{2n} P(j)(2n-j) = 2n-\overline{QL} \qquad (4.4)$$

where $\overline{QL}$ is the mean queue length of the main queue. Assume that an area of $\underline{M}$ units is required to store each request in the feedback queue in case of $p=u_0/(u_1M)$. The actual feedback queue area required by one of the feedback queues in Fig.4-1 is obtained by dividing Eq.(4.4) by $\underline{n}$ and multiplying by $\underline{M}$.

$$\text{Actual feedback queue area} = (2-\overline{QL}/n)M \qquad (4.5)$$

Idle time probability of one of the feedback queues in Fig.4-1 is approximately equal to the probability that no requests (i) exist in the assembled feedback queue in Fig.4-2, so that

$$\sum_{j=1}^{n} P(n+j)\ {}_{n-1}C_{j-1} / {}_nC_j = (n-1)!(np)^n \sum_{j=1}^{n} jp^j/(A(n-j)!) \ . \qquad (4.6)$$

Results of calculation are shown in Figs.4-3,4-4, and 4-5 by the dotted lines. In each figure, $1/u_0$ is equal to 110 msec and $1/u_1$ is assumed to be calculated by the following equation;

$$1/u_1 = 1/3(n-1/n) \times 10 + 12.5 \quad \text{(msec)} . \qquad (4.7)$$

Eq.(4.7) is a result of the following model. Suppose that $\underline{n}$ points are in line. If transition occurs from one point to another, its required time is expressed as in Eq.(4.7), where values 10 and 12.5 are the mean transition time between adjacent points and staying time at a point, respectively. These special parameters are used just because the mean service rates are expressed like those in the configuration of the multiple speech output system

described below . The parameter M is varied from 1 to 5. Given M, u_0 and u_1 , the queueing process is assumed to be a Poisson type.

Idle time probability of the feedback queue is shown in the dotted lines of Fig.4-3. It remarkably increases from certain points of n. Effects of M on the idle time probability are quite large, but gradually saturate . Fig.4-4 shows the actual feedback queue area which decreases at a remarkable rate with the increase of the idle time probability shown in Fig.4-3. Fig. 4-5 shows the availability of the main queue. It gradually saturates to 100 % and becomes too overloaded to serve any more multiplicity.

Consider the points of n in Fig.4-3 where the idle time probability shows a rapid increase for M=a.An actual feedback queue area of nearly 2(a+1) would be required

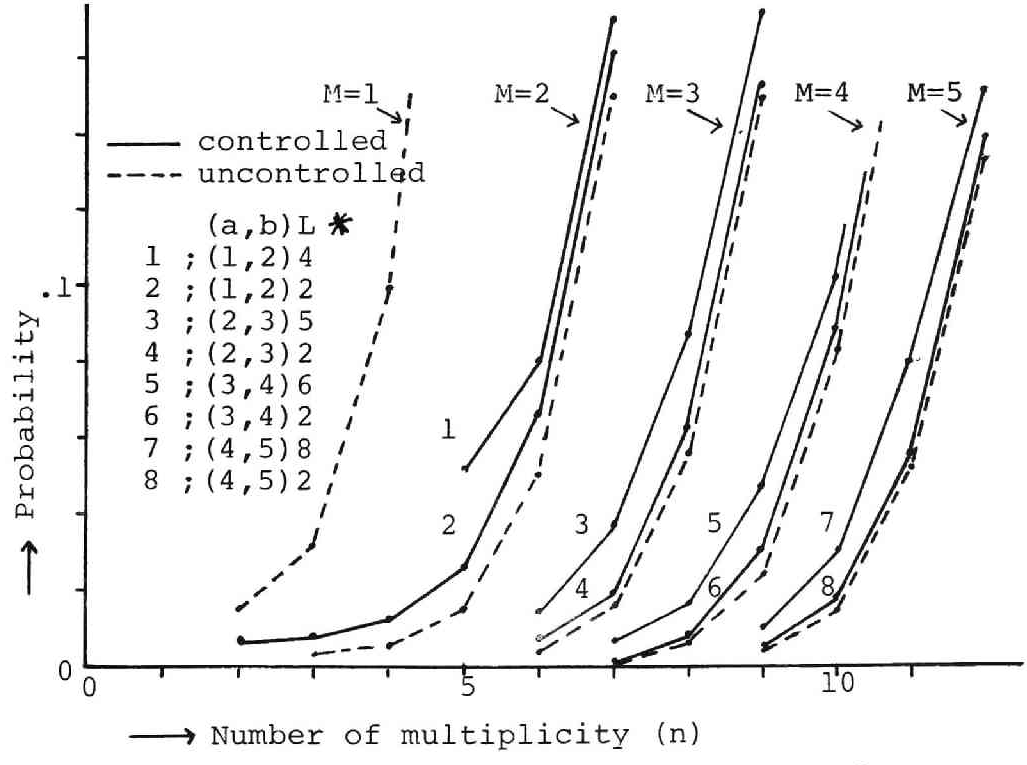


Fig. 4-3 Idle time probability of the feedback queue

* see p.105

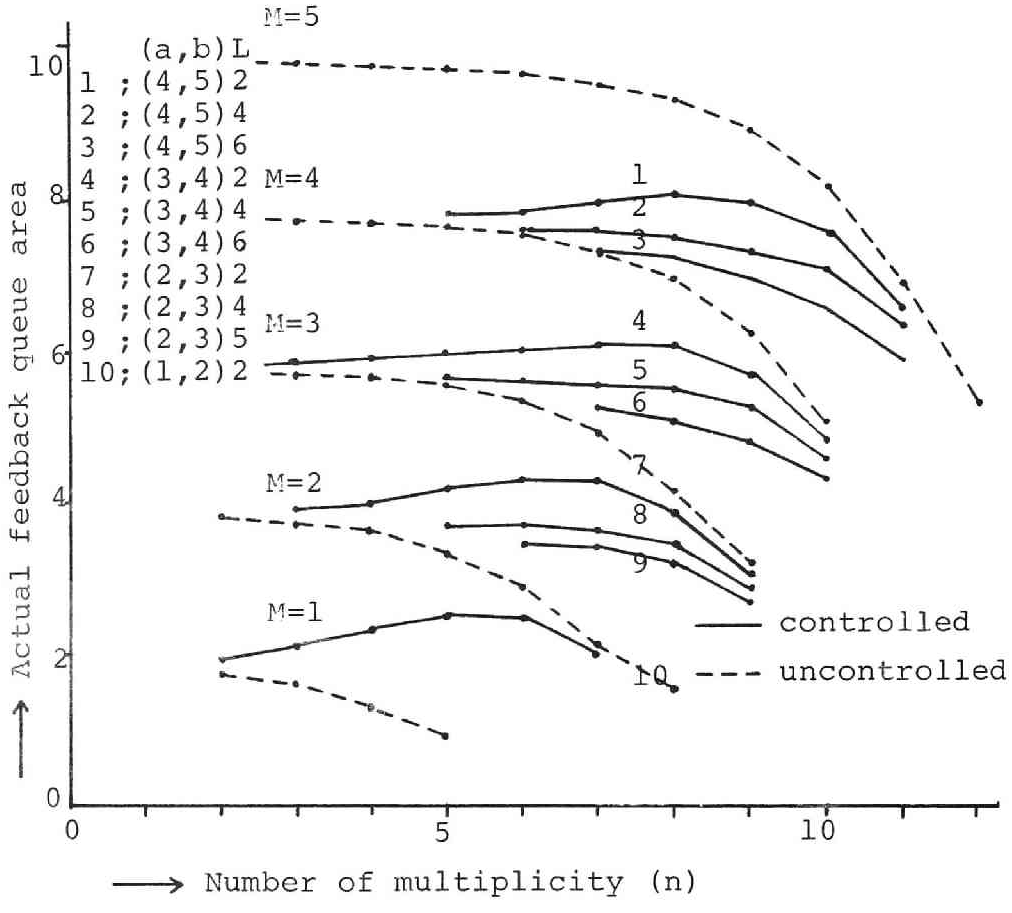


Fig. 4-4 Actual feedback queue area

in order to achieve low probability of the idle time of the feedback queue at those points of n.

Now, the controlled queueing process will be given. A control scheme proposed here is as follows:
Suppose that M=a is normally in use. But, if the queue length of the main queue exceeds a threshold value L, then the service rate of the feedback queue is reduced to ku_0/a, in order to suppress the excessive arrivals of requests to the main queue. This k takes the value a/b where a and b are positive integer values (b>a).
By adopting this scheme it would be expected that the idle time probability of the feedback queue between two

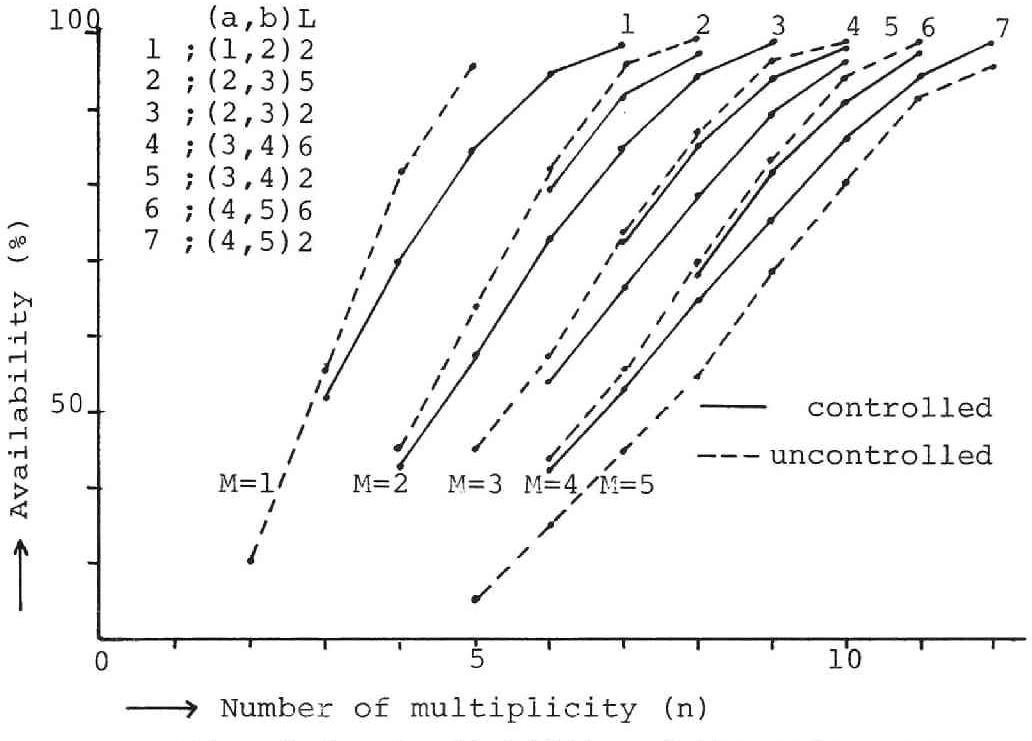


Fig. 4-5 Availability of the main queue

points of $\underline{n}$, where it shows rapid increase for M=a, but almost zero for M=(a+1), will be about zero at the cost of a smaller increase of the actual feedback queue area.

In this controlling scheme, the arrival rate to the main queue is nu_0/a if $0 \leqslant j < L$, nu_0k/a if $L \leqslant j \leqslant n$, otherwise $(2n-j)ku_0/a$. $\underline{L}$ is set smaller than $\underline{n}$ because a value greater than $\underline{n}$ has no effect on reducing the idle time probability of the feedback queue. The equations to describe the process are obtained in the same manner as in Eq.(4.1).

$$u_1 P(1) = nu_0/a \; P(0) \qquad ; \; j=0$$

$$u_1 (P(j+1)-P(j)) = nu_0/a (P(j)-P(j-1)) \qquad ; \; 0<j<L$$

$$u_1 P(L+1) - (nku_0/a+u_1) P(L) + nu_0/a P(L-1) = 0 \qquad ; \; j=L$$

$$u_1 (P(j+1)-P(j)) = nku_0/a (P(j)-P(j-1)) \qquad ; \; L<j \leqslant n$$

$$u_1 P(j+1) - ((2n-j)ku_0/a + u_1)P(j) + \qquad (4.8)$$
$$(2n-j+1)ku_0/a P(j-1) = 0 \quad ; \; n<j<2n$$
$$u_1 P(2n) = ku_0/a P(2n-1) \quad ; \; j=2n$$

These equations can be solved easily in an iterative manner as follows.

$$P(j) = (np)^j / A \quad ; \; 0 \leqslant j < L$$
$$P(j) = k^{j-L} (np)^j / A \quad ; \; L \leqslant j \leqslant n \qquad (4.9)$$
$$P(j) = n^n n! k^{j-L} p^j / (A(2n-j)!) \quad ; \; n+1 \leqslant j \leqslant 2n$$

$$A = \sum_{j=0}^{L} (np)^j + \sum_{j=L+1}^{n} k^{j-L} (np)^j + n!(nkp)^n k^{-L} \sum_{j=1}^{n} ((kp)^j / (n-j)!) \qquad (4.10)$$

where $p = u_0/(u_1 a)$. The following are obtained as in the previous section:

Availability of the main queue = $1 - 1/A$. (4.11)

The assembled feedback queue area to store a request is,

$$(\sum_{j=0}^{L-1} P(j) + 1/k \sum_{j=L}^{2n} P(j))a \quad . \qquad (4.12)$$

Assembled queue length is,

$$\sum_{j=0}^{2n} P(j)(2n-j) = 2n - \overline{QL} \quad . \qquad (4.13)$$

Then the mean actual feedback queue area in Fig.4-1 is obtained by multiplying Eq.(4.12) and (4.13) and dividing by n.

Actual feedback queue area

$$= (2 - \overline{QL}/n)(\sum_{j=0}^{L-1} P(j) + 1/k \sum_{j=L}^{2n} P(j))a \qquad (4.14)$$

where $\overline{QL}$ means the mean queue length of the main queue. The idle time probability of the feedback queue in Fig.4-1 is approximately calculated as follows.

Idle time probability of the feedback queue

$$= ((n-1)!k^{n-L}(np)^n \sum_{j=1}^{n} j(kp)^j/(n-j)!)/A \qquad (4.15)$$

These are plotted in Figs.4-3,4-4, and 4-5. Here the notation of (a,b)L means that the queueing model is under control in such a manner that if the queue length of the main queue is less than $\underline{L}$, then M=a, otherwise M=b. In each figure, $1/u_0$ is set to be equal to 110 msec and $1/u_1$ is assumed to be calculated by Eq.(4.7).

The idle time probability of the feedback queue in Fig.4-3 shows explicitly the effects of the control scheme. It remains almost zero over the points of $\underline{n}$ where it would show a rapid increase if M=a and the system was uncontrolled. But, it approaches to the curve of the idle time probability in the case where M=a+1 would be in use. Fig.4-4 shows the actual feedback queue area which inclines to approach to the curve in the case where M=a+1. These graphs indicate that the control scheme is effectively operating on the model. For example, consider the case of (2,3)5. Fig. 4-3 shows that the idle time probability in the case of the uncontrolled model where M=2 is about 0.05 at n=6. But, this control scheme reduces it to 0.015 or so. Furthermore, the actual feedback queue area remains near the curve corresponding to the case where M=2 rather than M=3, as described in Fig.4-4. In Fig.4-5, the availability of the main queue is shown. This indicates that the load of the main queue is reduced a little as compared with the case of the uncontrolled model.

4.3 Controlled Queueing Model of the Multiple Speech Output System

4.3.1 System Configuration

The multiple speech output system in which the queueing control is implemented is somewhat different in the configuration from the system described in both Chapter II and III. One of the main differences between them is related to the methods of editing input sentences and storing data on the magnetic disk. The system described here stores pre-edited dyad segment sequences for input sentences of a user in advance in a cylinder of the magnetic disk. This is done by using the method of Chapter II. A dyad segment sequence is stored over several record areas, the last of which is kept so as not to be packed. It is only because of simplicity for subsequent processings.

Output double buffers are also allocated to each user as in the system described in Chapter II. They can, however, store several units of dyad segment sequences.

The segment transfer is repeated several times until one of the output double buffers becomes empty. Then it is requested that some units of the next dyad segment sequences be transferred from the pre-assigned area on the magnetic disk into the required output double buffers. But, when many active users are receiving speech outputs, these requests will often form a queue. If the processing is late, then the unexpected pause will interrupt the output speech sounds (queueing error rate).

4.3.2 Controlled Queueing Process

It is a problem to decide how many units of dyad segment sequences should be read out at one time of disk accessing. It is closely related to the queueing error rate, and the necessary size of output double buffers. If large units of them were read out, the queueing error rate would be suppressed to almost zero up to a certain number of multiplicity, but, the required buffer size would result in a large capacity. While, if small units were read out, the queueing error rate would show too rapid an increase for a maximum number of multiplicity to be achieved sufficiently.
Then, it would be expected to control the whole system in order to reduce the queueing error rate at the cost of smaller increase of the output double buffers.

Now, the correspondence is considered between the queueing model described in Fig.4-1 and the multiple speech output system presented here. The system is approximate and globally abstracted to form a cyclic queueing model. The main queue in Fig.4-1 corresponds to the queue waiting for accessing the magnetic disk. The mean value of its service time is almost equal to the mean access time of the magnetic disk, where each active user is assumed to occupy each different cylinder.
The values 10 and 12.5 in Eq.(4.7) are the seek time to the adjacent cylinder and the mean rotation time of the disk, respectively.

Feedback queues in Fig.4-1 whose size is limited to a capacity of two, are corresponding to the output double buffers. The mean value of their service time is about 110 x M msec, where 110 msec is the mean value of the actual duration of one dyad segment sequence and M is

the number of units of dyad segment sequence read out from the magnetic disk at one time of accessing. Entering the main queue occurs when one of the output double buffers becomes completely empty.

The queueing error rate defined in Chapter II is the idle time probability of the feedback queue. Since CPU spent time is the ratio of the time in which CPU is occupied by the speech production to the total time, it is corresponding to the availability of the main queue. Actual feedback queue area defined in Eq.(4.14) corresponds to the size of the output double buffers used by a user.

Queueing control of the system is done as follows: Unless the system would be operated under some control, the occasional overload or underload conditions of the system would occur due to the random arrival of requests. But, if the real-time monitor of the system could forecast the overload conditions, it could forbid a coming request before arrival. This control of requests before their arrivals would be easily executed by reading out a longer length of data units from the magnetic disk. When one more dyad segment sequence is read out, the arrival to the main queue will be prolonged by one more T_F, because the interval during which the read data is in the output double buffers is increased by as much as T_F . While the required size of the output double buffers is increased by L_F to store longer data. In other words, this control scheme can take care of requests more during idle time periods, and prevent too many requests from occurring in a busy period. This scheme is illustrated schematically in Fig.1-1 in Section 1-2.

But, this control scheme is not directly analogous to the model shown in Fig.4-1. In that model, the service rate of the feedback queues is immediately degraded

at the time when the length of the main queue exceeds $\underline{L}$. However, in the speech output system, there is a certain time lag before the service rate of the feedback queue is degraded, because the read data may be kept waiting at least once in one of the output double buffers. This is one of the main differences between the model and the actual system. But, as a preliminary consideration, results of analysis of mathematical model described in Section 4.2 are revealing and available for qualitative estimation of the effects of the control scheme.

4.3.3 Method of Simulation of the System

A system simulation in real-time mode is conducted in the same manner as in Chapter III, in order to evaluate the performance of the system and investigate the effects of the control scheme for system improvement.

Input sentences of each active user are punched in Roman letters whose length is about several hundreds of characters. They are read into the computer to be converted by the speech production program shown in Chapter II, into the corresponding dyad segment sequences. The resulting sequences of OXI data are stored on each cylinder area allocated to each active user. In order to simulate the output process of the output double buffers, two tables are stored residently in the core memory and referred to. They contain time intervals (quantized in 10 msec) and necessary buffer size of the dyad segment sequences read out at one disk accessing. At every 10 msec, an interrupt signal reduces by one the value

of the balance of time in one of the output double buffers. When it becomes zero, a request is set in the queue.

A support program would present the total amount of buffers used, the distribution of interval time when the queueing error continues, and so forth.

4.4 Results of the simulation

4.4.1 Uncontrolled Queueing Model

The results of the simulation are shown in Figs.4-6, 4-7 and 4-8 where M is fixed to a constant. Fig.4-6 shows the relationship between the queueing error rate and the number of multiplicity n. The error rate increases at a sharp rate for certain points of n. As expected from the preliminary consideration, effects of M are quite large on reducing the queueing error rate, but they saturate gradually with the increase of M. The maximum number of multiplicity is increased by 3 or more every time the value of M is increased by one.
The maximum numbers of multiplicity for M=1,2,3,and 4 are perhaps 4,7,10, and 13 respectively as long as the resulting unexpected pause gracefully degrades the output speech sounds.

Fig.4-7 shows the necessary size of the total output double buffers for each number of multiplicity up to which the queueing error rate remains quite small.

Fig.4-8 shows the probability distribution of the queue length of the main queue where M=2. The mean queue

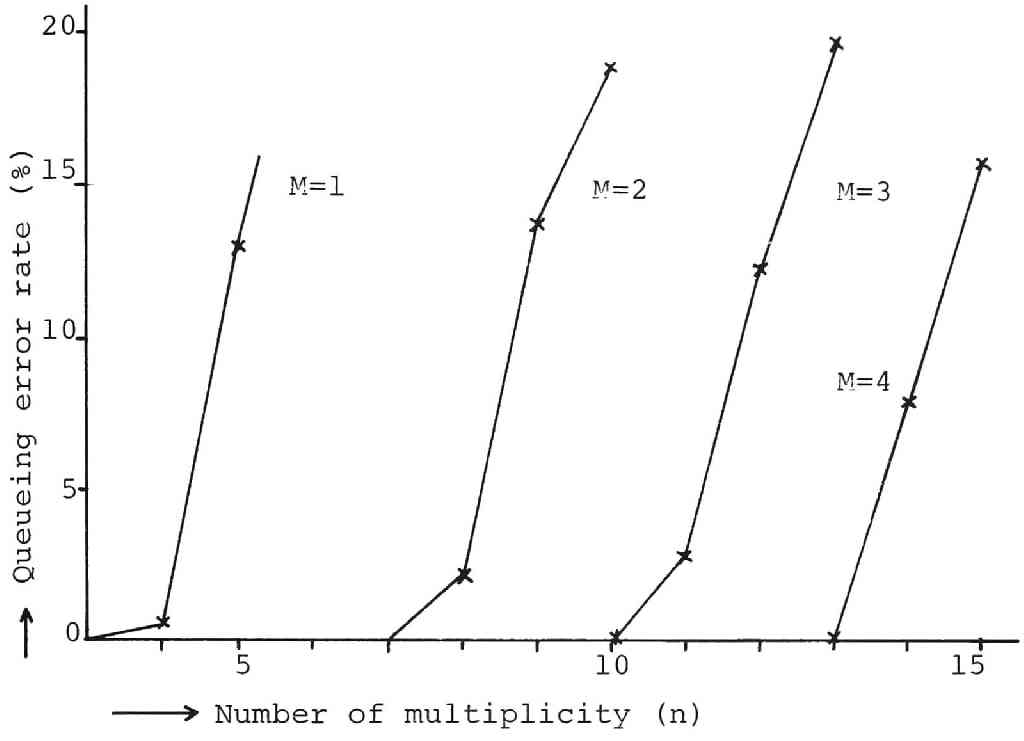


Fig. 4-6 Queueing error rate (uncontrolled)

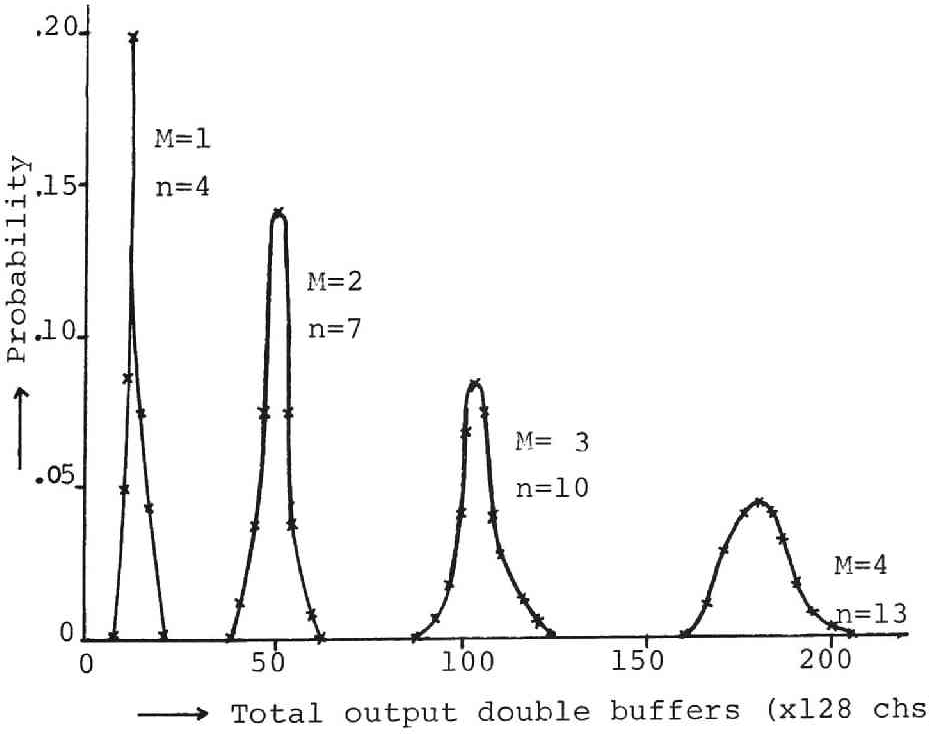


Fig. 4-7 Probability of the size of the total output double buffers (uncontrolled)

length is about 1.5 at the point n=7, but it increases at high rate, even at the point n=8. Queue length is found out to be extremely sensitive for a small increase of n.

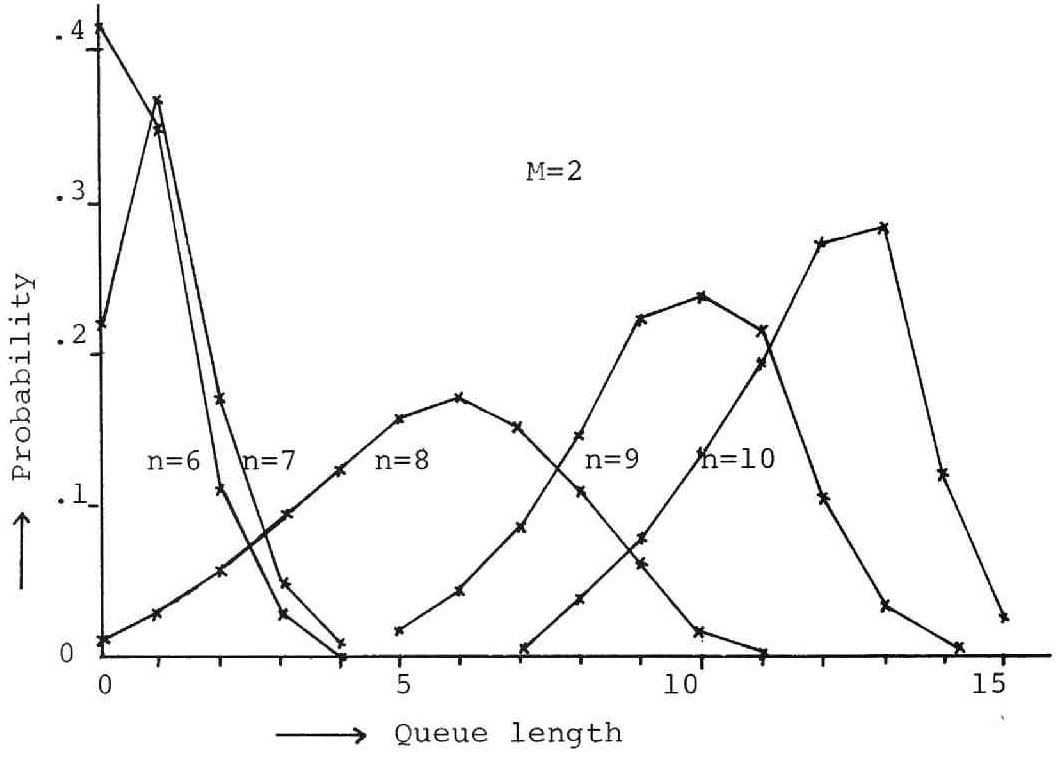


Fig. 4-8 Probability distribution of the queue length of the main queue (uncontrolled)

4.4.2 Controlled Queueing Model

Fig.4-9 shows the relationship between the number of multiplicity n and the queueing error rate. The notation (a,b)L means that the system is under control of the following scheme: The real-time monitor reads out a units of the dyad segment sequences from the disk, if the queue length of the main queue is less than L , otherwise it does b units. From the results of simulation described in Section 4.4.1, the parameter values for a would be selected to be 1, 2 and 3 for each number of multiplicity which lies among 5~7, 8~10, and 11~13 respectively. This is because it is sufficient enough for the effectively operating control scheme to take such a value of a in order to decrease the queueing error rate among those points of multiplicity. As the value L decreases, the queueing error rate is reduced sharply. While, the effects of increase of the parameter b is not so large when L is set to a rather small value.

Fig.4-10 shows the size of the total output double buffers used to produce independent speech sounds. This indicates the availability of the control scheme adopted in the system. Take for instance the special case, when the system is uncontrolled (M=2) . The probability distribution of the size of the total output double buffers used for producing 8 numbers of multiplicity is shown by the graph 1 in Fig.4-10. The mean size of the total output double buffers is about 53 x 128 chs. The corresponding queueing error rate seems very large which is shown in Fig.4-6 .

On the other hand, if M=3, then the queueing error rate is almost zero up to n=10 as shown in Fig.4-6. And the total buffer size for n=8 is shown in Fig.4-10 as the

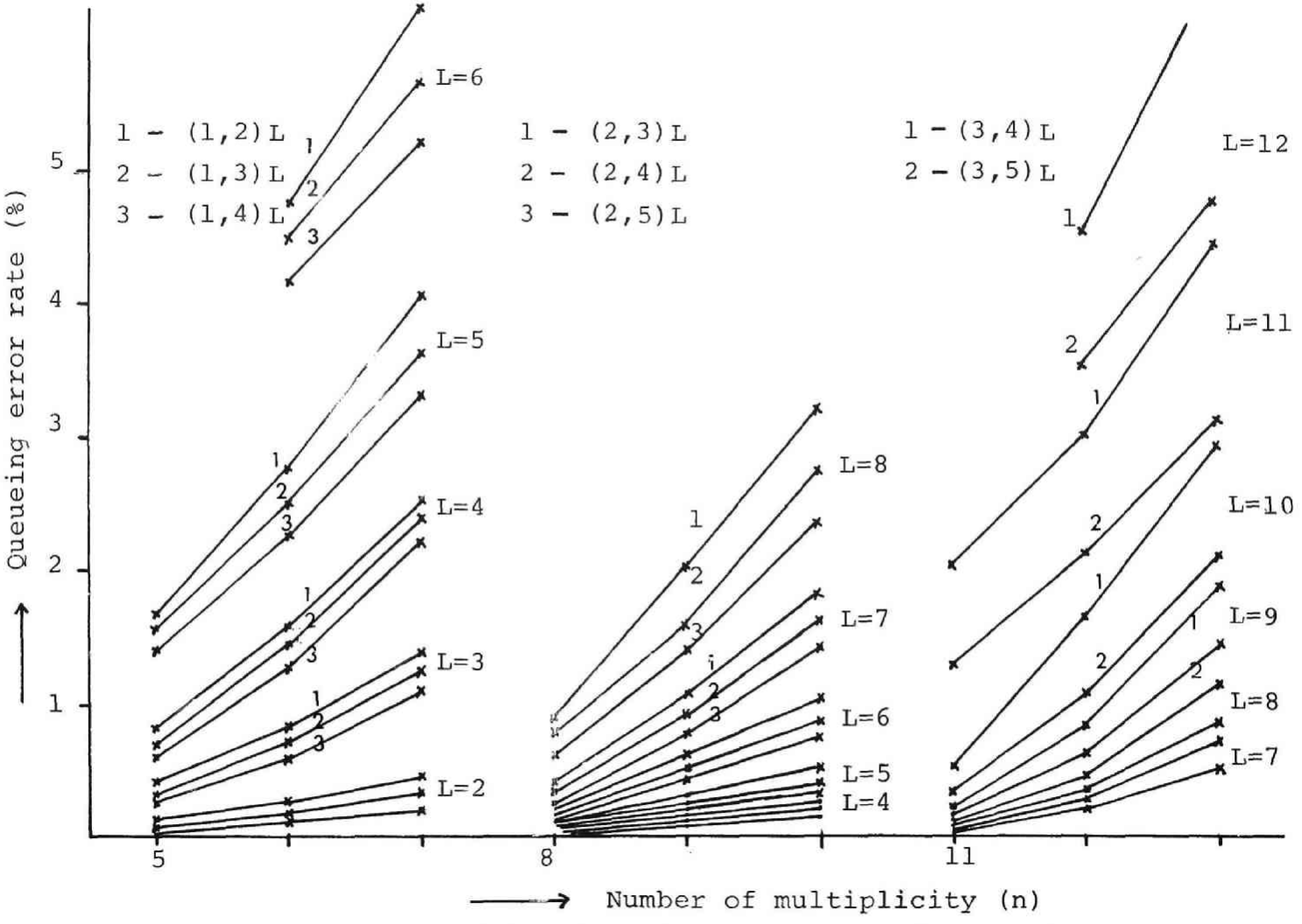


Fig. 4-9 Queueing error rate (controlled)

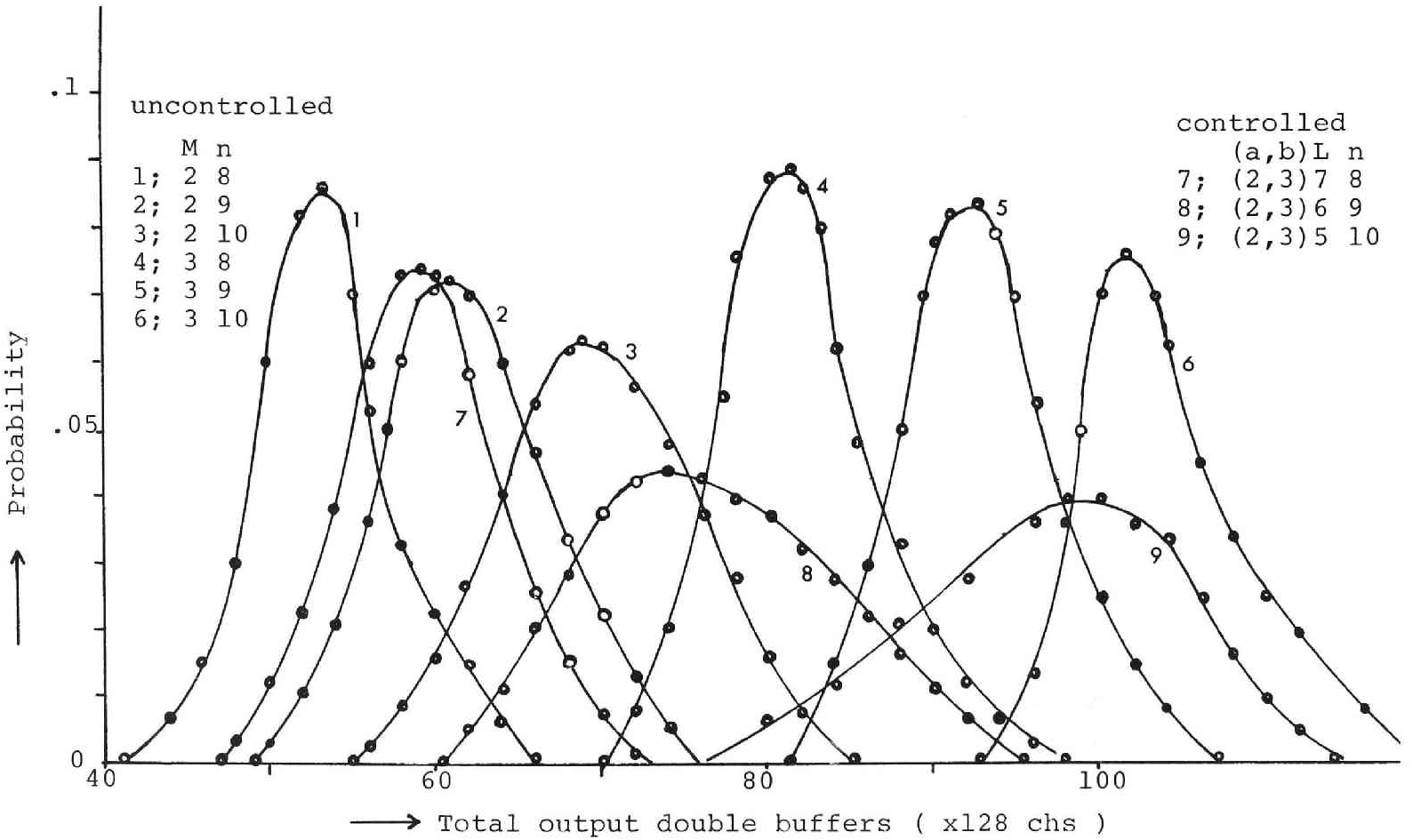


Fig. 4-10 Probability of the size of the total output double buffers

graph 4. The mean size amounts to about 81 x 128 chs. The graph 7 in Fig.4-10 shows the results of a controlled queueing model. The parameter (a,b)L is set to (2,3)7 and $\underline{n}$ is 8. The mean capacity of the total output double buffers is only 58 x 128 chs for n=8. However, the queueing error rate is quite decreased as shown in Fig.4-9. The increase of buffer size of 5 x 128 chs has almost the same effect on decreasing the queueing error rate as the greater buffer size would be used in the case where M=3 and n=8. But, where $\underline{n}$ is near to 7,10, and 13 for a=1,2 and 3, the effects of control are decreased and almost the same buffer size, as in the case of the larger value of $\underline{M}$, is required.

4.5 Conclusion

In this chapter, one of the examples of control problems of the queueing process was discussed. The objective system considered here, was assumed to be the multiple speech output system. The control scheme controlled requests before their arrivals to the system, judging from the waiting queue length and succeeded in keeping the system from the occasional overloaded conditions.

Among various issues on queueing control problems, the above scheme seemed to be somewhat innovative because the system was effectively controlled by using the close relationship between the input and output processes of the cyclic queue. Actually, the control scheme showed some

system improvement of the multiple speech output system.

In order to investigate the effects induced by the scheme, mathematical analysis and the system simulation in real-time mode were both presented. An analytical model was simplified and approximated to give insights on the qualitative estimation of the system. On the other hand, the system simulation in real-time mode was executed to discuss better effects of the control scheme adopted in the system.

The results of simulation showed that the control scheme was effective so that the system could produce much more independent speech outputs at the cost of smaller increase of the buffer size.

But, this is only a special case where simple queueing control is effectively applied. Of course, there are a wide variety of large computers running that need queueing control to utilize all of their resources. Quantitative considerations on the effective control schemes for them seldom appear because the controlled systems are too complicated for an analysis to be possible. Further research on this problem can be expected to help system designing and/or improve the system performance.

V Conclusion

In this thesis, outline of the multiple speech output system, its system evaluation and related queueing problems have been presented.

In Chapter II was described the experimental system producing simultaneous speech outputs (10 or more) of arbitrary Japanese messages. The output speech sounds were not so natural, but had enough intelligibility for human listening. Each synthesizer was implemented economically with only 180 IC gates. Furthermore, it could be easily connected to any type of computer for effective man-machine communication. In Chapter II were also discussed the load effect of the computer and some system's bottleneck points.

In Chapter III, the results of the system evaluation on the multiple speech output system were described. The powerful processing methods were also considered by which the maximum number of simultaneous outputs was raised to some extent. A technique of software monitoring was used to gather some statistical data occurring in the system. The simulation by simple devices and programs created environmental conditions under which many active users were connected to the computer.

The results of simulation showed that the simple remodeling of the processing methods made the total system be improved in its performance.

In Chapter IV, one of the examples of control problems of the queueing process was discussed. The objective system

considered was the multiple speech output system. The control was done by reducing arrival rate to the system when the overload conditions of the system could be forecast in advance. The results of the simulation indicated that the control scheme was effective so that the system could produce much more independent speech outputs at less cost. But, this is only a special case where simple queueing control is operating in the actual system. In general, a queueing control is an open problem for further intensive research.

This thesis has hoped to contribute to the progress of information science and technology. Recent IC technology has been miraculously developed so that system designing and implementation is not so difficult. This system can be constructed of IC assembly most easily and inexpensively and connected as a convenient speech output device in effective man-machine communication where speech output is indispensable. System evaluation is based on the special performance criteria developed throughout this paper. But, these evaluation techniques may become one of the steps to designing better systems.

BIBLIOGRAPHY

(1) R.H.Buron : "Generation of a 1000 word Vocabulary for Pulse Excited Vocoder Operating as an Audio Response Unit", IEEE Trans. AU-16,1 p.21 (1968)

(2) S.Miki,et al. : "Seat Reservation by Push Phones", Joint Convention of IPSJ, p.287 (Dec. 1972)

(3) J.L.Flanagan : "Synthetic Voices for Computers", IEEE Spectrum, p.22 (Oct. 1970)

(4) K.Nakata : "Speech Output Systems",J.of IECEJ,51,11 p.1427 (1968)

(5) L.H.Lee and R.B.Mulvany : "Now a Talking Computer Answers Inventory Inquiries", Electronics p.30 (Aug. 1963)

(6) Y.Koizumi,et al. : "Audio Response Unit in DIALS", Joint Convention of IPSJ, p.205 (Dec. 1972)

(7) F.Itakura : "New speech Analysis-Synthesis Method 'PARCOR'", Nikkei Electronics 2-12,p.58 (1973)

(8) N.R.Dixon and H.D.Maxey : "Terminal Analog Synthesis of Continuous Speech Using the Diphone Method of Segment Assembly", IEEE Trans. AU-16, p.40 (1963)

(9) Y.Kato and K.Ochiai : "Terminal Analog Speech Synthesizer", Joint Convention of the Acous.Soc. of Japan, I-2-14 (Nov. 1966)

(10) L.Rabiner : "Speech Synthesis by Rule", BSTJ 47,1 p.17 (1968)

IECEJ ; The Institute of Electronics and Communication Engineers of Japan

IPSJ ; Information Processing Society of Japan

(11) E.Matsui,et al. : "Dynamic Analog Speech Synthesizer", Joint Convention of the Acous. Soc. of Japan 2-2-9 (Nov. 1965)
(12) K.Nakata and T.Miura : "A Method of Speech Synthesis for Multiplexed Audio Response", J. of IECEJ 52-C,10 p.579 (1969)
(13) E.Matsui : "A Multiple Speech Output System by Pitch Synchronous Compilation of Vocal Units", Joint Convention of IECEJ 1-3-13 (Nov. 1968)
(14) T.Sakai and Y.Niimi : "Speech Synthesis by Zero-crossing Wave", Joint Convention of IECEJ,p.161(1967)
(15) T.Sakai and K.Ohtani : "A Study of Speech Synthesis System Using Zero-crossing Wave", J.of IECEJ 52-C,11 p.704 (1969)
(16) K.Ohtani : "Speech Synthesis and Recognition by Computer", Doctoral Thesis in Kyoto Univ.,(July 1972)
(17) T.L.Saaty : "Elements of Queueing Theory", McGraw-Hill p.12 (1961)
(18) T. Homma : "Theory of Queues", Rikogakusha , p.21 (1966)
(19) T.Ono : "SYDAS II", Joint Convention of IPSJ, p.139 (Dec. 1972)
(20) H.Kitagawa : "Evaluation on Large Scale Computer Systems", Doctoral Thesis in Kyoto Univ. (Feb. 1972)
(21) M.Yadin and P.Naor : "On Queueing Systems with Variable Service Capacities", Naval Research Logistics Quarterly 14,7 p.43 (1967)
(22) U.Narayan : "A Controlled Transportation Queueing Process", Management Science 16,7,p.446 (March 1970)
(23) R.F.Gebhard : "A Queueing Process with Bilevel Hysteretic Service Rate Channel", Naval Research Logistics Quarterly 14,7 p.55 (1967)

(24) Y.Ohno,et al. : "Basic Idea for Actual Time Processing for Mars 101 Unit Seat Reservation System", Hitachi Technical Report 46,6,p.95 (1964)

(25) W.W.Chu ; "Demultiplexing Considerations for Statistical Multiplexors", IEEE COM-20,3,p.603 (1972)

(26) D.W.Davies : "The Control of Congestion in Packet Switching Networks", IEEE COM-20,3,p.546 (1972)

(27) T.Sakai,K.Ohtani and S.Tomita : "Multiple Speech Output System by Zero-crossing Wave", Technical Report of the Professional Group on Automaton & Information Theory of IECEJ (June 1970)

(28) T.Sakai,K.Ohtani and S.Tomita : "Multiple Speech Output System by Zero-crossing Wave", Joint Convention of IECEJ S-3-10 (1970)

(29) T.Sakai,K.Ohtani and S.Tomita : "On-line,Real-time, Multiple Speech Output System", Proceedings of IFIP Congress 1971, Ljubliana TA-4,p.686 (1971)

(30) T.Sakai,K.Ohtani and S.Tomita : "On-line,Real-time Multiple Speech Output System", J. of IECEJ 55-D,3 p.149 (1972)

(31) J.C.R.Licklider and I.Pollack : "Effects of Differentiation, Integration and Infinite Peak Clipping upon the Intelligibility of Speech",JASA 20,1,p.42 (1948)

(32) J.C.R.Licklider ; "The intelligibility of Amplitude-dichotomized,Time-quantized Speech Wave", JASA 22, 6 p.820 (1950)

(33) R.W.A Scarr : "Zero-crossings as a Mean of Obtaining Spectral Information in Speech Analysis", IEEE Trans. AU-16,p.257 (1968)

(34) S.E.G.Öhman : "Coarticulation in VCV Utterences Spectrographic Measurements", JASA 39,1,p.151 (1966)

(35) K.Ohtani : "Private letters", (1973)
(36) T.Sakai and S.Tomita : "On-line,Real-time Multiple Speech Output System and its System Evaluation", Joint Convention of IPSJ p.151 (1972)
(37) T.Sakai and S.Tomita : "On-line,Real-time,Multiple Speech Output System and its System Evaluation", Technical report of Professional Group on Automaton & Language of IECEJ (Jan. 1973)
(38) T.Sakai,K.Ohtani and S.Tomita : "On-line,Real-time Multiple Speech Output System and its System Evaluation", STUDIA PHONOLOGICA VI p.70 (1972)
(39) D.P.Gaver,et al. : "Probability Models for Buffer Storage Allocation Problem", JACM 18,2,p.186 (1971)
(40) M.A.Leibowitz : "An Approximate Method for Treating a Class of Multiqueue Problem", J. of IBM,p.204 (July 1961)
(41) L.Takacs : "Two Queues Attended by a Single Server", Operations Research 16,3,p.639 (1967)
(42) R.B.Cooper,et al. : "Queues Served in Cyclic Order", BSTJ,p.675 (March 1969)
(43) P.J.Denning : "Effects of Scheduling on File Memory Operations", AFIPS SJCC,p.9 (1967)
(44) S.Doshita : "Studies on the Analysis and Recognition of Japanese Speech Sounds", Doctoral Thesis in Kyoto Univ., p.102 (Sep. 1969)
(45) L.G.Roberts, et al. : "Computer Network Development to Achieve Resource Sharing", AFIPS SJCC, p.543 (1970)
(46) F.E.Heart, et al. : "The Interface Message Processor for the ARPA network", AFIPS SJCC, p.551 (1970)

APPENDIX

Deduction of Eq. (3.1) and Eq. (3.2) where m=2

(j) : The queue length in the main queue (state).

$P_t(j)$: Probability that the queue length in the main queue is just j at time t.

$P_{t+\Delta t}(j)$: Probability that the queue length in the main queue is just j at time $t+\Delta t$.

$P(j)$: Probability that the queue length in the main queue is just j in the equilibrium state.

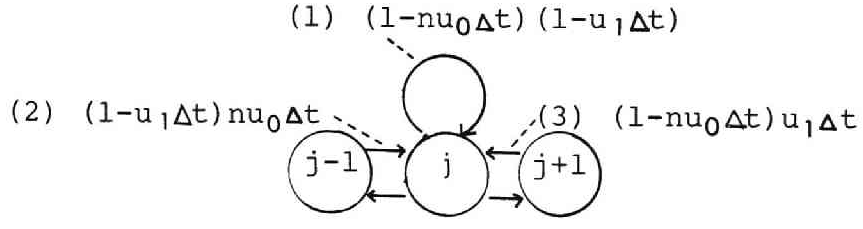
In the case $j \leqslant n$, state transition to the state j occurs, when

(1) Neither request is input/output to/from the main queue

(2) Only one request is input, but none output

(3) Only one request is output, but none input

This state transition is expressed as follows.



$P_{t+\Delta t}(j)$ is the summation of these disjoint probabilistic events and is expressed as follows.

$$\begin{aligned} P_{t+\Delta t}(j) = & P_t(j)(1-u_1\Delta t)(1-nu_0\Delta t) \\ & + P_t(j-1)(1-u_1\Delta t)nu_0\Delta t \\ & + P_t(j+1)(1-nu_0\Delta t)u_1\Delta t \end{aligned} \qquad \text{(A.1)}$$

Eq. (A.1) is devided by Δt, then

$$(P_{t+\Delta t}(j) - P_t(j))/\Delta t = -(u_1+nu_0)P_t(j)$$

$$+nu_0P_t(j-1)$$
$$+u_1P_t(j+1)+O(\Delta t)\,. \qquad (A.2)$$

As $t\to\infty$, left hand side of Eq. (A.2) becomes zero in the equilibrium state. This leads to the equation.

$$u_1(P(j+1)-P(j))=nu_0(P(j)-P(j-1))\,. \qquad (A.3)$$

In the case $j>n$, $P_{t+\Delta t}(j)$ is also expressed in the summation of the following disjoint probabilistic events.

(1) $(1-u_1\Delta t)(1-(2n-j)u_0\Delta t)P_t(j)$

(2) $(1-u_1\Delta t)(2n-j+1)u_0\Delta tP_t(j-1)$

(3) $(1-(2n-j-1)u_0\Delta t)u_1\Delta tP_t(j+1)$

Then,

$$\begin{aligned}P_{t+\Delta t}(j)=&(1-u_1\Delta t)(1-(2n-j)u_0\Delta t)P_t(j)\\&+(2n-j+1)(1-u_1\Delta t)u_0\Delta tP_t(j-1)\\&+(1-(2n-j-1)u_0\Delta t)u_1\Delta tP_t(j+1)\end{aligned} \qquad (A.4)$$

Eq. (A.4) is devided by Δt, then $t\to\infty$.
It results in

$$u_1P(j+1)-((2n-j)u_0+u_1)P(j)+(2n-j+1)u_0P(j-1)=0\,. \qquad (A.5)$$

Boundary conditions are obtained in the same way.

$$u_1P(1)=nu_0P(0) \qquad (A.6)$$
$$u_1P(2n)=u_0P(2n-1) \qquad (A.7)$$

From Eq. (A.3) and (A.6), $P(j)$ is for $j\leqslant n$,

$$P(j)=(np)^jP(0) \qquad (A.8)$$

where $p=u_0/u_1$.

$P(j)$ is assumed for $j>n$, to be

$$P(j)=Kp^j/(2n-j)! \ , \quad (K: \text{constant}) \tag{A.9}$$

and substituted into the left side of Eq. (A.5). Then it completely satisfies Eq. (A.5). $P(j)$ assumed in Eq. (A.9) is the solution for the case $(j>n)$.
From Eq. (A.7),(A.8) and (A.9), Eq. (3.2) is obtained.

LIST OF SYMBOLS

Chapter II

P	Wave element
aP^r	Segment
OXI	Zero-crossing interval
S.T.C.	Single tuned circuit
A/D	Analog to digital
f_1	First formant frequency
f_2	Second formant frequency
(V-C)	Vowel-Consonant context
(C-V)	Consonant-Vowel context
(V-V)	Vowel-Vowel context
chs	Characters(6 bits)
T-FF	T-flip flop
P-register	Register containing one digit of OXI
R-register	Register containing repetition parameter(r)
A-register	Register containing amplitude parameter(a)
$a_i(b_i)$	One of the areas of the external core memory allocated to the user (i)
R/W	read/write
$A_i(B_i)$	One of the output double buffers allocated to the user (i)
*	End marker of the segment
D/A	Digital to analog
D(k)	Numbered element in the set of dyad segment sequence
P(k)	Frequency in use of D(k) in the normal contexts
τ(k)	Actual duration of D(k)

T_F	Mean duration of the dyad segment sequence
	Mean arrival interval of the editing request
$X(k)$	Number of zero-crossings in produced speech sounds of $D(k)$
T_S	Mean duration of the segment
	Mean arrival interval of the segment transfer request
$H(k)$	Necessary memory capacity to store $D(k)$
K_1	Mean amount of controlling information
$M(k)$	Number of segments contained in $D(k)$
K_2	Mean number of zero-crossings in one second
L_F	Mean length of the dyad segment sequence
L_S	Mean length of the segment
n	Number of multiplicity
t_{ac}	Cycle time of the external core memory
$T_{R/W}$	Set-up-time for R/W switching
e_1	Ratio of memory cycles used by the data channel controller to the total time
C	Transfer rate of the data channel
e_2	Ratio of read mode in the external core memory
H_t	Operation time for the editing program
W_t	Operation time for the I/O control program
N	Number of the data channels

Chapter III

(j)	Queue length in the main queue
m	Limited queue size of the user queue
$P(j)$	Probability that the queue length in the main queue is just (j)
u_1	Mean service rate of the main queue
u_0	Mean service rate of the user queue
p	u_0/u_1

R	Number of records on each track of the secondary memory
W	Total access time required
k	Number of requests
T	Rotation time of the secondary memory
FIFO	First-in-first-out
SATF	Shortest-access-time-first
$(P_{10}+P_{01})$	Probability of transition from one cylinder to the other
S	Seek time of the magnetic disk
Sl_i (.....)	Table containing the duration of the successive dyad segment sequences
BS_i	Table indicating the amount of buffer size of the user (i) in use
BR_i	Table showing the time interval when the user buffer (i) does not become empty

Chapter IV

(j)	Queue length in the main queue
P(j)	Probability that the queue length in the main queue is just(j)
u_0/M	Mean service rate of the feedback queue
u_1	Mean service rate of the main queue
M	Positive integer value Number of units of dyad segment sequence read out from the magnetic disk
p	$u_0/(u_1M)$
$\overline{QL}$	Mean queue length in the main queue
L	Threshold value of the queue length of the main queue
a,b	Parameter value of M
k	a/b